CAI-BASIC

A PROGRAM TO TEACH THE PROGRAMMING LANGUAGE 'BASIC'

THOMAS ANTHONY BARRY

LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIF. 93940

LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIF. 93940

United States
Naval Postgraduate School

THESIS

CAI-BASIC

A Program to Teach the

Programming Language "BASIC"

by

Thomas Anthony Barry

Thesis Advisor A.B. Roberts

September 1971

Approved for public release; distribution unlimited.

LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIF. 93940

CAI-BASIC

A Program to Teach the

Programming Language "BASIC"

by

Thomas Anthony Barry
Lieutenant, United States Navy
B. S., United States Naval Academy, 1965

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 1971

B 34222
c.1

ABSTRACT

This paper presents a computer aided instruction program that fulfills the objectives of teaching a simple programming language, interpreting student responses, and executing and editing student programs. The CAI-BASIC program is written in FORTRAN IV, level G, and executes on IBM- 2741 terminals while running under the CP-67/CMS time sharing system on the U. S. Naval Postgraduate School's IBM-360/67 computer system. The instructional phase of CAI-BASIC presents the fundamentals of "BASIC," a simple user oriented language, in seven lessons. During the instructional sessions the student is presented material and, based on his response to questions, he is routed to the next sequence of instructions. The execution phase of CAI-BASIC allows execution of "BASIC" programs, and has an optional debug feature that provides a trace of program variables to aid the student in finding programming errors. In the event of programming errors the user may enter an edit mode to correct mistakes in his program.

TABLE OF CONTENTS

I. INTRODUCTION 4

A. DEFINITION OF CAI 4

B. CAI DEVELOPMENTS 4

C. OBJECTIVES 6

II. DESCRIPTION OF CAI-BASIC 7

A. INSTRUCTION PHASE 7

B. EXECUTION PHASE 8

III. LIMITATIONS AND EXTENSIONS 10

IV. CONCLUSION 12

APPENDIX A USING CAI-BASIC 13

APPENDIX B CAI-BASIC PROGRAM 16

APPENDIX C TERMINAL SESSION WITH CAI-BASIC 70

BIBLIOGRAPHY 119

INITIAL DISTRIBUTION LIST 120

FORM DD 1473 121

I. INTRODUCTION

During the last few years many sophisticated projects in Computer Assisted Instruction (CAI) have evolved. The potential of this area has captured the imagination of researchers and the public at large. Yet the U. S. COMMISSION ON INSTRUCTIONAL TECHNOLOGY (ref. 9) found that the status of instructional technology in American Education was low in both quantity and quality. There are many reasons that CAI has not been accepted with any marked degree of enthusiasm by educators in general. The major reasons are the exorbitant costs of individualized instruction and illusions over just what CAI is.

A. DEFINITION OF CAI

Computer Assisted Instruction is machine augmented instruction that differs from Programmed Instruction or simple page turning machines in that the computer system has logic and memory capabilities to assist in the instructional process. The main advantage in CAI is that it can individualize education. A student can proceed at his own pace, spending more time on difficult material, and quickly covering material which comes easy to him.

B. CAI DEVELOPMENTS

A. G. OETTINGER, in his book RUN, COMPUTER, RUN (ref. 6) states that educators and the computer industry are equally to blame for the failure of CAI to realize its potential. Instructional technology has been force fed,

oversold, and prematurely applied; and as a result the educators are wary of false promises. OETTINGER feels that colleges and universities will be an effective proving ground for future educational technology. However, at the present time, of the thousands of colleges and universities in the nation only twenty-five are considered major CAI centers.[1]

A common problem confronting the nation's colleges and universities is that of providing instruction in elementary computer programming.[2] The popularity of the computer science field, and the range of application of computers in everyday life means that the demand for people who know how to communicate with computers is growing rapidly. The solution to the problem is the computer itself. "Whatever the state of CAI with respect to other subjects, the computer is the ideal instrument for teaching its own use."[3]

W. R. SMITH and J. L. YOUNG, graduate students at the Naval Postgraduate School, presented a proposal for the use of computer Assisted Instruction at the Naval Postgraduate School (ref. 10). As this introduction has done, they created an awareness for the need of CAI, and they made specific recommendations to provide a balanced program for the entire community to emulate. In particular they recommended commencing student projects to probe the potentials of CAI

---

1. STOLUROW, L. M., "Computer Assisted Instruction," REPORT OF A CONFERENCE, U. S. Department of HEW, p. 49, 1969.

2. FENICHEL, R. R., WEIZENBAUM, J., AND YOCHELSON, J. C. "Program to Teach Programming," COMMUNICATION OF THE ACM, v. 13, p. 141, March 1970.

3. IBID., p. 141-142.

at the Naval Postraduate School.

It was with these ideas about using the computer to teach computer programming and providing a CAI base at the Naval Postgraduate School that CAI-BASIC was undertaken.

C. OBJECTIVES

The objective of CAI-BASIC was to develop a CAI program to teach a computer programming language on the Naval Postgraduate School's time-sharing terminals. The programming language chosen was "BASIC" (Beginners All-purpose Symbolic Instruction Code), a computer language developed at Dartmouth College in the mid-1960's. "BASIC" was selected because of its simplicity yet fairly large range of application for the user who wants to take advantage of computer processing. It was felt that "BASIC" would provide the average non-computer oriented graduate student with satisfactory results in a minimal amount of study time.

With this objective in mind three sub-goals were determined to be essential for the CAI-BASIC project: to be able to interpret student responses, to be able to execute "BASIC" programs in order to give the student sufficient programming practice, and to be able to allow editing of "BASIC" programs.

II. DESCRIPTION OF CAI-BASIC

The CAI-BASIC program is written in FORTRAN IV, level G, and executes on IBM-2741 terminals while running under CP-67/CMS Time Sharing System on the Naval Postgraduate School's IBM-360/67 Computer System. The "BASIC" language implemented in CAI-BASIC is standard non-extended "BASIC" (refs. 3, 5, 10).

The CAI-BASIC program consists of two phases; an instruction phase, and an execution and editing phase. A detailed discussion of the use of CAI-BASIC is presented in APPENDIX A.

A. INSTRUCTION PHASE

The instruction phase presents fundamental concepts common to most programming languages: identifiers, variables, iteration, branching, subroutines, built in functions, and recursion. The instruction set consists of seven lesson modules which, for reasons of coherency, are dependent upon each other. The underlying philosophy in preparing the lesson modules was that an important idea should be presented only after a need for it has clearly been established.[4]

Each lesson consists of instruction sequences followed by questions, evaluation of student responses, routing to the next sequence of instruction,

---

4. FENICHEL, OP. CIT., p. 142.

and a summary of the lesson followed by questions and/or problems to program. The student is allowed to progress through the lessons at his own speed, and allowed to review lessons or execute programs at the completion of each lesson.

Student responses are interpreted by one of two methods depending upon the type of question asked. When the question is of the true-false, multiple choice or give the answer type then the response is compared with a pre-stored result. When the student is asked to reply with a "BASIC" statement or to write a "BASIC" program, the CAI-BASIC compiler interprets the "BASIC" statements for syntactic correctness. When the student writes a program he can compare his answer with a pre-stored result, and if the answer is wrong the student is shown a solution program.

B. EXECUTION PHASE

The heart of the execution phase is the "BASIC" compiler that is used at the Naval Postgraduate School's Computer Center to execute "BASIC" programs in a batch mode. This compiler, written in FORTRAN IV, level G, was modified from a batch processing compiler to a line-by-line interpreter and incorporated into the CAI-BASIC program, and called the CAI-BASIC COMPILER.

The CAI-BASIC COMPILER has an added feature to aid students in debugging programs on the terminal. The added feature is called a "DEBUG" function and it produces a list of numeric and alpha-numeric data that were put into the program; and, in addition it produces a trace of all simple variables as they are assigned values during program execution.

At the completion of program execution or when an execution error occurs, the student has an option to enter the CAI-BASIC Edit mode to correct his program one line at a time. When all editing is completed the program is executed again.

III. LIMITATIONS AND EXTENSIONS

The major limitation of CAI-BASIC is the absence of any supervision over the student's progress as he proceeds through the instruction phase. A possible extension to this project would be to add a supervisory routine to maintain the student's progress and to keep a record of his errors. Thus, having the student's progress level and error record, CAI-BASIC could be tailored to instruct the student at his own learning level. In other words, CAI-BASIC could be made into a more completely interactive teaching program (refs. 1, 4).

An additional feature that would aid the student interaction is a communication link between the student and a professor so that student questions can be answered. R. R. FENICHEL (refs. 1, 2) described how students enter a special mode to type questions during the instruction session on the terminal. Then at the beginning of the student's next instruction session, all of his previous questions are answered on the terminal. FENICHEL refers to this as the "mailbox" system. Students type questions for the "mailbox" and the professor replys with answers or pertinent information for the "mailbox."

Another limitation to the existing CAI-BASIC program is the inability of CAI-BASIC to monitor the student when he enters the execution phase to write programs. The CAI-BASIC system does not oversee the student, beyond checking for syntax errors in his program. There is no method of inspecting his

programs from a tutorial point of view in order to help with semantic programming errors.

A foreseeable development to bring CAI-BASIC into the Artificial Intelligence field would be to make CAI-BASIC an "intelligent" tutor. This would remove the present inflexibility of interpreting student responses with prestored answers and open the possibility of CAI-BASIC understanding the logic of student responses and student programs.

IV. CONCLUSION

As stated in the objectives, the goal of CAI-BASIC was to develop a CAI program to teach a computer programming language on the Naval Postgraduate School's terminal system. The technical aim of completing the project and bringing it to an operating level has been met. However, an evaluation of the practicality of the project is still in the speculative stage.

Several computer and non-computer oriented students have tried CAI-BASIC and they found it to be both understandable and beneficial, but a full scale evaluation of the effectiveness of CAI-BASIC as a teaching tool is not possible from this limited sampling. It is sufficient to say that initial indications are that CAI-BASIC can provide a satisfactory means of learning a simple but capable programming language in a minimal amount of time. The student's participation in the learning experience encourages him to teach himself and to practice new programming skills. It is hoped that future use of CAI-BASIC by students will demonstrate its practicality.

Although the scope of CAI-BASIC was not particulary broad, the results obtained provide a basis for future CAI projects. The foundation has been laid, and the limitations and extensions of the previous section suggest a direction for future CAI efforts. Many previous CAI projects have suffered from too many or too large a scope of objectives, and as a result they never realize any practical results. This project has met its objectives and, hopefully, future CAI projects will benefit from its results.

APPENDIX A

In its present configuration the CAI-BASIC program occupies approximately ten cylinders of private disk space and requires an overlaying routine to execute under the control of CP-67/CMS which limits users to a 175K virtual machine. The overlaying routine, written by a Naval Postgraduate School Computer Center system programmer, initally loads only lesson 1 when CAI-BASIC is loaded into core. When any lesson other than lesson 1 is used, it is overlaid onto lesson 1.

The overlaying routine allows students to log on to CP-67/CMS as general users and to link into CAI-BASIC. This allows CAI-BASIC to be available to all of the terminal stations. The user gains access to CAI-BASIC by logging on the terminal as follows:

```
login yyyygxx          (xx is the terminal nr.)
                       (yyyy is your user nr.)
ENTER PASSWORD
npg
ENTER 4-DIGIT PROJECT NUMBER.... ETC.
0623cs04
READY AT (TIME) ON (DATE)
CP
link 0909p 191 192
ENTER PASSWORD
teach
SET TO READ ONLY
i cms
CMS..VERSION 01/21/71
login 192 t,p
**T (192) READ ONLY**
teach
```

Small print is typed in by the user, and the capitalized print is the computer terminal response.

The CAI-BASIC program consists of two phases, an instruction phase and an execution and editing phase. After logging on to the system, the user enters the main routine, CAIBAS, which directs the general flow of CAI-BASIC.

If the user has not used CAI-BASIC before he enters the instruction phase, he is presented with an introduction to CAI-BASIC, and then is sequentially guided through the lessons. After each lesson the user is given the opportunity to terminate his session, to review a lesson, to enter the execution phase, or to go on to the next lesson.

If the user has used CAI-BASIC before, he is given the opportunity to review any lesson or to go to the next lesson in his learning sequence.

When the execution and editing routine, TEST1, is entered the user is allowed to execute standard "BASIC" programs (refs. 3,5,10). The Test1 routine incorporates the "BASIC" compiler, COMPLR, which interprets each "BASIC" input statement. Each "BASIC" statement is input one line at a time; and, if a syntax error occurs on the input, COMPLR prints an error message. The user can then input the correct "BASIC" statement. When the "end" statement is input, the program is checked for global errors; and, if none occur, the program is executed.

When global errors or execution errors occur, the user is given the choice of entering the edit mode of TEST1 to correct his program, or using the "DEBUG" feature to find his programming errors.

The edit mode allows the user to delete, add to, or correct "BASIC"

statements in his program one line at a time. The user is given a listing of his program with reference numbers for editing purposes. To edit a program the user is first asked for the statement reference number, and then asked for the specific edit command.

A statement is deleted by typing in the reference number of the statement to be deleted, hitting the carriage return, and then typing in the command "DEL."

A statement is added 'after' the reference number typed in by typing in the command "ADD1" followed by the "BASIC" statement to be added. To add a statement before the first statement in the program, the reference number zero (0) is used.

A statement is corrected by typing in the reference number of the statement to be corrected, hitting carriage return, and then typing in the 'entire' correct "BASIC" statement.

The "DEBUG" feature is designed to provide useful information to the user who has encountered an execution error in his program. It is used by adding the key word "DEBUG" as a statement in the user's program.

The "DEBUG" feature gives the user a list of all numeric and alpha-numeric data that were used in the program; and, in addition, it produces a trace of all simple variables with their values as they are assigned values during program execution.

CAI-BASIC : A PROGRAM TO TEACH THE PROGRAMMING LANGUAGE "BASIC"

T.A. BARRY

U.S.NPS. 8/17/71

THE FILES USED IN CAI-BASIC UNDER CP-67/CMS ARE AS FOLLOWS :

MAIN PROGRAM---- CAIBAS

SUBROUTINES----INITAL
CRUNCH
TEST1
LESON1--> LESON7

OVERLAYING ROUTINES FOR CP-67/CMS :

SEARCH ALOAD EXIT

THE FOLLOWING SUBROUTINES ARE PART OF THE CAIBASIC COMPILER , AS MODIFIED FROM THE NPS "BASIC" COMPILER USED FOR BATCH PROGRAMS, BUT ARE NOT INCLUDED AS PART OF THE CAI-BASIC PROGRAM :

ACON	COMPLR	ERR	INSNO
APRINT	CONV	DUMMY	LTL
BUFFIL	CPRINT	EVAL	RANF
CEND	DIM	EXEC	TEST

SYMBOL TABLE : (GLOBAL SYMBOLS AND VARIABLES)

CARD	VECTOR HOLDING STUDENT INPUT
CARDP	VECTOR HOLDING STUDENT INPUT WITH BLANKS REMOVED
ILNGTH	LENGTH OF INPUT IN VECTOR CARDP
ASTRSK	WHEN CARDP(1)=ASTRSK THE USER HAS MADE A TYPING ERROR
ALPHA	VECTOR HOLDING THE LETTERS OF THE ALPHABET
DIGIT	VECTOR HOLDING THE DIGITS 0-->9
ALPHA(25)	THE LETTER 'Y'
ALPHA(14)	THE LETTER 'N'

```
C                                                                       CAI00010
CC     CAIBAS IS THE MAIN PROGRAM .IT IS A COMPUTER AIDED INSTRUCTION
CC     PROGRAM WHICH TEACHES FUNDEMENTAL PROGRAMMING CONCEPTS USING     CAI00030
CC     THE LANGUAGE BASIC. THE HEART OF THE PROGRAM PROVIDES INSTRUCTION CAI00040
CC     AND INTERPRETES STUDENT RESPONSES. SOME ADDED FEATURES ARE       CAI00050
CC     EXECUTION AND EDITING OF 'BASIC' PROGRAMS IN "TEST1"             CAI00060
CC     SUBROUTINES CALLED FROM CAIBAS ARE:
CC        INITAL
CC        CRUNCH
CC        TEST1
CC        LESON1 --> LESON7
CC        SEARCH
CC        ALOAD
CC
CC     SYMBOL TABLE : (LOCAL SYMBOLS AND VARIABLES)
CC        LSNUM     CURRENT LESSON NUMBER
CC        OLD       FLAG . SET TO 1 IF USER HAS USED CAI-BASIC BEFORE
CC        LES1 --> LES7       VARIABLES FOR OVERLAYING ROUTINES
C                                                                       CAI00070
                                  COMMON                                CAI00080
     -          STACK(100), PROG(2000), CARD(80), CARDP(80), ALPHA(48),  CAI00090
     -          IAPTR, INPTR, IADATA(500) , XNDATA(500) , STRING(5),     CAI00100
     -          DIGIT(10), IPRITB(10), LISTST(100), ISTLST(100),         CAI00110
     -          PRT(2500), NERRS, INST, NSTLST, DEBUG, DOLSGN, QUOTE,    CAI00120
     -          EQUALS, PARRT, DECMAL, PLUS, CMINUS, SLASH, COMMA,       CAI00130
     -          PARLFT, ASTRSK, BLANK                                    CAI00140
      COMMON  INTERP,IEXERR                                             CAI00150
      REAL*8 LES1/'LESON1  '/
      REAL*8 LES2/'LESON2  '/
      REAL*8 LES3/'LESON3  '/
      REAL*8 LES4/'LESON4  '/
      REAL*8 LES5/'LESON5  '/
      REAL*8 LES6/'LESON6  '/
      REAL*8 LES7/'LESON7  '/
      CALL SEARCH(LES1,N1)
      CALL INITAL                                                       CAI00160
      LSNUM=0
      OLD=0                                                             CAI00180
      WRITE(6,99)                                                       CAI00190
   99 FORMAT('0',' HI , WELCOME TO CAI-BASIC . THERE ARE ONLY A FEW '/' CAI00200
     *SIMPLE RULES TO REMEMBER IN ORDER TO HAVE A SUCCESSFUL SESSION'/' CAI00210
     *ON THE TERMINAL WITH CAI-BASIC :                                 '// 5X,'
     * 1. WHEN ASKED FOR A RESPONSE , TYPE IN THE CORRECT REPLY AND '/' CAI00230
     *HIT THE CARRIAGE RETURN KEY ON THE RIGHT SIDE OF THE KEYBOARD.'// CAI00240
     * 5X,' 2. IF YOU MAKE A TYPING ERROR WHILE MAKING ANY RESPONSE '/'
```

```
     *OR INPUT , TYPE IN FOUR DOLLAR SIGNS ($$$$) AFTER THE ERROR OR'/' CAI00260
     *ANYWHERE ON THAT INPUT LINE AND HIT CARRIAGE RETURN .THE      '/' CAI00270
     *ENTIRE LINE WILL THEN BE IGNORED AND YOU CAN TYPE IN THE CORRECT'/CAI00280
     *' INPUT OR RESPONSE .                                    '// 5X,'
     * 3. IF AT ANY POINT IN THE SESSION YOU WANT TO STOP THE SESSION'/'CAI00300
     * TYPE IN THE WORD ''QUIT'' AS SOON AS YOU ARE ASKED FOR THE NEXT'/CAI00310
     *' RESPONSE ,HIT CARRIAGE RETURN,THEN HIT ATTN KEY AND TYPE LOGOUT
     *. '                                                      // 5X,'
     *4. DURING YOUR TERMINAL SESSION CAI-BASIC WILL HALT OCCASIONALLY'/CAI00330
     *' TO LET YOU READ A SEQUENCE OF INFORMATION . WHEN YOU ARE    '/' CAI00340
     *READY TO CONTINUE , TYPE IN ''GO'' , AND HIT CARRIAGE RETURN'// 5X
     *,' 5. DURING YOUR TERMINAL SESSION YOU MAY NOTICE THAT        '/CAI00360
     *' THE TYPING IS NOT ALWAYS PERFECT . SOME DAYS THE COMPUTER IS'/' CAI00370
     *NOT UP TO PAR AND YOU WILL HAVE TO ADJUST TO THE MINOR IRRITANT'//
     *'    IF YOU ONLY WANT TO EXECUTE PROGRAMS AT THIS TIME THEN    '/' CAI00390
     *REPLY : YES  ; OTHERWISE REPLY : NO   .'//)                     CAI00400
   1  READ(5,101,END=300)CARD                                        CAI00410
      CALL CRUNCH(ILNGTH)                                           CAI00420
      IF(CARDP(1).EQ.ASTRSK) GO TO 1
      IF(CARDP(1).NE.ALPHA(25)) GO TO 2                             CAI00430
C                                                                   CAI00440
CC    EXECUTION ONLY PHASE                                          CAI00450
C                                                                   CAI00460
      CALL TEST1                                                    CAI00470
      WRITE(6,109)
  400 READ(5,101,END=401)CARD
      CALL CRUNCH(ILNGTH)
      IF(CARDP(1).EQ.ALPHA(25)) RETURN
      IF (CARDP(1).EQ.ALPHA(14)) GO TO 402
  401 WRITE(6,104)
      GO TO 400
   2  IF(CARDP(1).EQ.ALPHA(14)) GO TO 3                             CAI00490
  300 WRITE(6,104)                                                  CAI00500
      GO TO 1                                                       CAI00510
    3 WRITE(6,100)                                                  CAI00520
  100 FORMAT('0',' IF THIS IS YOUR FIRST SESSION WITH CAIBASIC, THEN'/' CAI00530
     *REPLY:  YES  ;OTHERWISE REPLY:  NO  .'//)                     CAI00540
    4 READ(5,101,END=302)CARD                                       CAI00550
  101 FORMAT(80A1)                                                  CAI00560
      CALL CRUNCH(ILNGTH)                                           CAI00570
      IF(CARDP(1).NE.ALPHA(14))GO TO 20                             CAI00580
C                                                                   CAI00590
CCC   OLD USER. DETERMINE PROGRESS LEVEL AND ROUTE TO DESIRED LESSON
C
                                                                    CAI00620
  402 OLD=1                                                         CAI00630
      WRITE(6,102)                                                  CAI00640
  102 FORMAT('0','          NOW YOU MAY CHOOSE TO REVIEW ANY LESSONS'/' CAI00650
```

```
     *OR TO BEGIN WHERE YOU ENDED DURING YOUR LAST SESSION.')                CAI00660
      WRITE(6,105)                                                          CAI00670
  105 FORMAT('0',10X,'THE LESSONS ARE AS FOLLOWS :    '//10X,' LESSON 1 CAI00680
     *PROGRAM FORMAT AND BASIC DEFINITIONS
     *                          '/10X,' LESSON 2 REMARKS,INPUT/OUTPUT
     * AND DATA'/10X, ' LESSON 3 ASSIGNMENT STATEMENTS AND BUILT IN FUNCCAI00700
     *TIONS'/10X,' LESSON 4 BRANCHING '/11X,'LESSON 5 LOOPING AND SUBSCR
     *IPTED VARIABLES'/10X,' LESSON 6 SUBROUTINES AND RECURSION '/10X,'
     *LESSON 7 SUMMARY OF BASIC STATEMENTS
     *   '/'*** WHEN YOU ARE READY TO CONTINUE , TYPE IN GO AND HIT '/' CAI00730
     *THE CARRIAGE RETURN ***'//)                                           CAI00740
      READ(5,101,END=5)CARD                                                 CAI00750
    5 WRITE(6,103)                                                          CAI00760
  103 FORMAT('0', 5X,' REPLY WITH THE NUMBER OF THE LESSON YOU WISH TO C
     *OVER'//)                                                             CAI00780
    6 READ(5,101,END=301)CARD                                               CAI00790
      CALL CRUNCH(ILNGTH)                                                   CAI00800
      DO 7  I=2,8
      IF (CARDP(1).EQ.DIGIT(I)) GO TO 9                                     CAI00820
    7 CONTINUE                                                              CAI00830
  301 WRITE(6,104)                                                          CAI00840
  104 FORMAT('0','*** YOUR REPLY WAS TYPED  INCORRECTLY; CHECK THE  '/' CAI00850
     *QUESTION AND REPLY AGAIN ***'//)                                      CAI00860
      GO TO 6                                                               CAI00870
    9 LSNUM=I-1
   10 GO TO (11,12,13,14,15,16,17),LSNUM
   11 CALL LESON1                                                           CAI00900
      GO TO 25                                                              CAI00910
   12 CALL ALOAD(LES2,N1)
      GO TO 25                                                              CAI00930
   13 CALL ALOAD(LES3,N1)
      GO TO 25                                                              CAI00950
   14 CALL ALOAD(LES4,N1)
      GO TO 25                                                              CAI00970
   15 CALL ALOAD(LES5,N1)
      GO TO 25                                                              CAI00990
   16 CALL ALOAD(LES6,N1)
      GO TO 25                                                              CAI01010
   17 CALL ALOAD(LES7,N1)
      GO TO 25                                                              CAI01030
C                                                                           CAI01040
C     INSTRUCTION PHASE  ** NEW USER
C                                                                           CAI01060
   20 IF(CARDP(1).EQ.ALPHA(25))GO TO 21                                     CAI01070
  302 WRITE(6,104)                                                          CAI01080
      GO TO 4                                                               CAI01090
   21 WRITE(6,106)                                                          CAI01100
  106 FORMAT('0',10X,'CAIBASIC IS A PROGRAM TO TEACH YOU THE        '/' CAI01110
```

```
     *FUNDAMENTALS OF A PROGRAMMING LANGUAGE.THE LANGUAGE TO BE LEARNED'CAI01120
     */' IS BASIC ; A SIMPLE LANGUAGE FOR THE USER WHO HAS LITTLE'/' KNOCAI01130
     *WLEDGE OF COMPUTERS AND WHOSE PRIMARY INTEREST IS IN OBTAINING '/'
     *RESULTS.'/10X,'THE SIMPLICITY OF THE BASIC LANGUAGE AND ITS RANGE'CAI01150
     */' OF CAPABILITIES SHOULD ALLOW YOU TO LEARN THE LANGUAGE AND '/' CAI01160
     *WRITE PROGRAMS IN A MINIMAL AMOUNT OF TIME.'//5X,' THE REFERENCE T
     *EXTS RECOMMENDED FOR CAIBASIC ARE :'//5X,' 1. BASIC LANGUAGE MANUA
     *L , TN # 0211-12 APRIL 1971'/' ( FREE UPON REQUEST IN I-147 )'//5X
     *,' 2. INTRODUCTION TO COMPUTING THROUGH THE BASIC LANGUAGE , R.L.
     *NOLAN'/' (BOOKSTORE / MAIN LIBRARY)'//5X,' 3. BASIC PROGRAMMING ,
     *V.C. HARE'/' (COMPUTER CENTER LIBRARY )    '//'****  AFTER YOU HAVCAI01170
     *E FINISHED READING AN INPUT , TYPE IN GO AND HIT THE RETURN KEY'/'CAI01180
     *  AND THE PROGRAM WILL CONTINUE ***'//)                          CAI01190
      READ(5,101,END=307)CARD                                         CAI01200
  307 WRITE(6,107)                                                    CAI01210
  107 FORMAT('0',10X,'DURING YOUR TERMINAL SESSION YOU WILL BE LEARNING'CAI01220
     */' THE STRUCTURE OF THE LANGUAGE BASIC. THE INSTRUCTION SET    '/' CAI01230
     *CONTAINS 7 LESSONS AND YOU MAY PROCEED THROUGH THE LESSONS AT '/' CAI01240
     *YOUR OWN SPEED. '//)                                            CAI01250
      WRITE(6,105)                                                    CAI01260
      READ(5,101,END=308)CARD                                         CAI01270
  308 WRITE(6,108)                                                    CAI01280
  108 FORMAT('0',10X,' IN EACH LESSON YOU WILL BE GIVEN INSTRUCTION '/' CAI01290
     *SEQUENCES AND THEN YOU WILL BE ASKED QUESTIONS TO SEE IF YOU  '/' CAI01300
     *UNDERSTOOD THE INSTRUCTIONS.THE QUESTIONS WILL BE OF VARIOUS  '/' CAI01310
     *TYPES: MULTIPLE CHOICE,TRUE/FALSE,ACTUAL PROGRAM STATEMENTS,ETC.'/CAI01320
     *' YOU WILL BE PROMPTED FOR YOUR ANSWER, AND WHEN READY TYPE IN '/'
     *YOUR RESPONSE AND HIT THE RETURN KEY.'//10X,'IF YOU KEEP THE TELETCAI01340
     *YPE OUTPUT FROM YOUR TERMINAL SESSION'/' YOU WILL HAVE A READY REFCAI01350
     *ERENCE FOR FUTURE USE'//)                                        CAI01360
      READ(5,101,END=309)CARD                                         CAI01370
  309 LSNUM=1                                                         CAI01380
   24 IF (LSNUM.GT.7) GO TO 40
      GO TO (11,12,13,14,15,16,17),LSNUM
   25 WRITE(6,109)                                                    CAI01410
  109 FORMAT('0',10X,' DO YOU WANT TO TERMINATE YOUR INSTRUCTION SESSIONCAI01420
     * ?'/'  (REPLY : YES OR NO )'//)                                 CAI01430
   26 READ(5,101,END=310)CARD                                         CAI01440
      CALL CRUNCH(ILNGTH)                                             CAI01450
      IF(CARDP(1).EQ.ALPHA(25)) GO TO 40                              CAI01460
      IF (CARDP(1).EQ.ALPHA(14)) GO TO 30                             CAI01470
  310 WRITE(6,104)                                                    CAI01480
      GO TO 26                                                        CAI01490
C     OLD USERS PICK NEXT LESSON ; NEW USERS GET NEXT LESSON IN SEQUENCECAI01500
   30 WRITE(6,209)                                                    CAI01510
  209 FORMAT('0',10X,' IF YOU WANT TO EXECUTE PROGRAMS AT THIS TIME '/' CAI01520
     *THEN REPLY : YES ; OTHERWISE REPLY : NO AND THE INSTRUCTION   '/' C
     *WILL CONTINUE .'//)                                             CAI01540
```

```
   32 READ(5,101,END=33)CARD                                                CAI01550
      CALL CRUNCH(ILNGTH)                                                   CAI01560
      IF(CARDP(1).EQ.ALPHA(14)) GO TO 34                                    CAI01570
      IF(CARDP(1).NE.ALPHA(25)) GO TO 33                                    CAI01580
C
CC    ENTER EXECUTION PHASE
C
      CALL TEST1                                                            CAI01590
      GO TO 25                                                              CAI01600
   33 WRITE(6,104)                                                          CAI01610
      GO TO 32                                                              CAI01620
   34 IF(OLD.EQ.1) GO TO 5                                                  CAI01630
      WRITE(6,110)                                                          CAI01640
  110 FORMAT('0',10X,' DO YOU WANT TO REVIEW A LESSON BEFORE GOING ON ?'CAI01650
     */' REPLY : YES OR NO'//)
   27 READ(5,101,END=311)CARD                                               CAI01670
      CALL CRUNCH(ILNGTH)                                                   CAI01680
      IF(CARDP(1).EQ.ALPHA(25)) GO TO 5                                     CAI01690
      IF (CARDP(1).EQ.ALPHA(14)) GO TO 31                                   CAI01700
  311 WRITE(6,104)                                                          CAI01710
      GO TO 27                                                              CAI01720
   31 LSNUM=LSNUM+1                                                         CAI01730
      GO TO 24                                                              CAI01740
C     USER INSTRUCTION SESSION COMPLETED.                                   CAI01750
   40 CONTINUE                                                              CAI01760
      STOP                                                                  CAI01770
      END


      SUBROUTINE TEST1                                                      TES00010
C                                                                           TES00020
CC    THIS ROUTINE ALLOWS EXECUTION AND EDITING OF BASIC PROGRAMS           TES00030
CC                                                                          TES00040
CC    SUBROUTINES CALLED FROM TEST1 ARE:                                    TES00050
CC       INITAL                                                             TES00060
CC       CRUNCH                                                             TES00070
CC       COMPLR                                                             TES00080
CC                                                                          TES00090
CC    SYMBOL TABLE : (LOCAL SYMBOLS AND VARIABLES)                          TES00100
CC                                                                          TES00110
CC       NEW       FLAG . SET TO 1 WHEN USER HAS EXECUTED FIRST PROGRAM    TES00120
CC       IEND      FLAG . SET TO 1 WHEN 'END' STATEMENT READ               TES00130
CC       IEXERR    FLAG . SET TO 1 WHEN AN EXECUTION/GLOBAL ERROR OCCURS   TES00140
CC       NERRS     FLAG . SET TO 1 WHEN A SYNTAX ERROR OCCURS IN INPUT     TES00150
CC       INTERP    FLAG . SET TO 1 WHEN ONLY INTERPRETING                  TES00160
CC       NSTMT     NUMBER OF PROGRAM STATEMENTS INPUT                      TES00170
CC       SFILE     SOURCE FILE VECTOR HOLDING PROGRAM STATEMENTS           TES00180
C        NREF      PROGRAM STATEMENT REFERENCE NUMBER                      TES00190
```

```
C         KTK       COUNTER TO DETERMINE IF REFERENCE NUMBER IS 1 OR 2   TES00200
CC                  DIGITS                                              TES00210
CC        IARRAY    VARIABLE USED IN CAI-BASIC COMPILER                 TES00220
CC        DEBUG          "    "   "    "     "                          TES00230
CC        OLD       FLAG . SET TO 1 WHEN USER HAS EDITED FIRST PROGRAM   TES00240
CC        INBIG          "    "   "    "     "                          TES00250
CC        NFOR           "    "   "    "     "                          TES00260
C                                                                       TES00270
                              COMMON                                    TES00280
     -          STACK(100), PROG(2000), CARD(80), CARDP(80), ALPHA(48),  TES00290
     -          IAPTR, INPTR, IADATA(500) , XNDATA(500) , STRING(5),     TES00300
     -          DIGIT(10), IPRIOR(10), LISTST(100), ISTLST(100),         TES00310
     -          PRT(2500), NERRS, INST, NSTLST, DEBUG, DOLSGN, QUOTE,    TES00320
     -          EQUALS, PARRT, DECMAL, PLUS, CMINUS, SLASH, COMMA,       TES00330
     -          PARLFT, ASTRSK, BLANK                                    TES00340
      COMMON  INTERP,IEXERR                                             TES00350
      DIMENSION SFILE(100,80)                                           TES00360
C                                                                       TES00370
CC    EXECUTION PHASE                                                   TES00380
CC                                                                      TES00390
CC                                                                      TES00400
CC    IF NEW SET DON'T PRINT INTRO AGAIN                                TES00410
C                                                                       TES00420
      IF(NEW.EQ.1)GO TO 3                                               TES00430
      WRITE(6,99)                                                       TES00440
   99 FORMAT('0',15X,' ****  CAIBASIC COMPILER  ****                  '//' TES00450
     *THE CAIBASIC COMPILER IS A LINE BY LINE INTERPRETER .THE BASIC'/'  TES00460
     *INTERPRETER ACCEPTS STANDARD BASIC STATEMENTS , AND IT ANALYZES'/'TES00470
     * EACH BASIC STATEMENT AS IT IS INPUT . AN ADDED FEATURE OF    '/' TES00480
     *THE CAIBASIC COMPILER IS AN EDITING AND A DEBUGGING ROUTINE   '/' TES00490
     *THAT ALLOWS THE USER TO ADD , DELETE AND CORRECT STATEMENTS ; '/' TES00500
     *AND TO GET A LISTING OF ALPHA-NUMERIC AND NUMERIC DATA USED , '/' TES00510
     *AND A TRACE OF ALL SIMPLE VARIABLES AS THEY ARE ASSIGNED VALUES'/'TES00520
     * IN THE PROGRAM.                                          '/10X,' TES00530
     *THE DEBUG FEATURE  IS USED BY ADDING THE KEY WORD  DEBUG     '/' TES00540
     *AS A STATEMENT TO YOUR PROGRAM .                         '/10X,' TES00550
     *THE EDITING MODE IS AVAILABLE TO THE USER WHEN AN EXECUTION  '/' TES00560
     *ERROR OCCURS AND AFTER SUCCESSFUL EXECUTION .            '/5X,' TES00570
     *  IN THE EVENT OF AN INPUT ERROR THE INTERPRETER WILL ANALYZE '/' TES00580
     * THE ERROR AND PRINT AN ERROR MESSAGE. IF THIS OCCURS FIND THE'/' TES00590
     *ERROR ,AND INPUT THE CORRECT BASIC STATEMENT FOR THE CURRENT  '/' TES00600
     *LINE OF INPUT .                                              '//' TES00610
     *   (NOTE : TYPING ERRORS CAN BE DELETED BY TYPING FOUR DOLLAR '/' TES00620
     *SIGNS ($$$$) ON THE SAME LINE AS THE ERROR , HITTING CARRIAGE '/' TES00630
     *RETURN , AND INPUTTING THE LINE AGAIN .)'//)                     TES00640
      OLD=0                                                             TES00650
    3 WRITE(6,300)                                                      TES00660
  300 FORMAT('0',10X,' **** CAIBASIC EXECUTION ****'//)                 TES00670
```

```
      CALL INITAL                                                       TES00680
      NEW=1                                                             TES00690
      DEBUG=C.0                                                         TES00700
      NSTMT=0                                                           TES00710
      IEND=0                                                            TES00720
      INBIG=1                                                           TES00730
      IARRAY=808                                                        TES00740
      NFOR=0                                                            TES00750
      INTERP=0                                                          TES00760
      IEXERR=0                                                          TES00770
    5 WRITE(6,100)                                                      TES00780
  100 FORMAT('0',5X,' INPUT BASIC PROGRAM NOW (ONE LINE AT A TIME )'///)TES00790
   10 READ(5,101,END=5)CARD                                             TES00800
  101 FORMAT(80A1)                                                      TES00810
      CALL CRUNCH(ILNGTH)                                               TES00820
      IF(CARDP(1).EQ.ASTRSK) GO TO 10                                   TES00830
C                                                                       TES00840
CC    FIND ' END ' STATEMENT                                            TES00850
C                                                                       TES00860
      DO 699 I=1,80                                                     TES00870
      IF(CARDP(I).NE.ALPHA(5).OR.CARDP(I+1).NE.ALPHA(14))GO TO 699      TES00880
      IF(CARDP(I+2).EQ.ALPHA(4).AND.CARDP(I+3).EQ.BLANK)GO TO 701       TES00890
  699 CONTINUE                                                          TES00900
  700 CALL CCMPLR(NFOR,IARRAY,ILNGTH,INBIG)                             TES00910
C                                                                       TES00920
CC    IF NO ERRORS , FILE INPUT STATEMENT                               TES00930
C                                                                       TES00940
      IF(NERRS.EQ.0) GO TO 9                                            TES00950
      IF(IEXERR.EQ.1)GO TO 9                                            TES00960
      NERRS=0                                                           TES00970
      GO TO 10                                                          TES00980
C                                                                       TES00990
CC    PUT IN END STATEMENT                                              TES01000
C                                                                       TES01010
  701 NSTMT=NSTMT+1                                                     TES01020
      IEND=1                                                            TES01030
      DO 702 I=1,80                                                     TES01040
  702 SFILE(NSTMT,I)=CARD(I)                                            TES01050
      GO TO 700                                                         TES01060
    9 IF(NSTMT.GT.99) GO TO 511                                         TES01070
      IF(IEND.EQ.1) GO TO 12                                            TES01080
       NSTMT=NSTMT+1                                                    TES01090
      DO 11 J=1,80                                                      TES01100
   11 SFILE(NSTMT,J)=CARD(J)                                            TES01110
       GO TO 10                                                         TES01120
  511 WRITE(6,512)                                                      TES01130
  512 FORMAT('0',' YOU HAVE REACHED THE MAXIMUM PROGRAM SIZE ALLOWED'/' TES01140
     *UNDER CAI-BASIC . YOU WILL HAVE TO MODIFY YOUR PROGRAM TO     '/' TES01150
```

```
     *STAY BELOW 100 PROGRAM STATEMENTS .'//)                                  TES01160
      GO TO 513                                                               TES01170
C                                                                             TES01180
CC    EXECUTION ERROR GO TO EDIT ROUTINE                                      TES01190
C                                                                             TES01200
   12 IF(IEXERR.EQ.1)GO TO 31                                                 TES01210
  513 WRITE(6,205)                                                            TES01220
  205 FORMAT('0',10X,' DO YOU WANT TO EDIT YOUR PROGRAM '/'   REPLY : YE
     *S  OR NO .'//)
  312 READ(5,101,END=315)CARD                                                 TES01250
      CALL CRUNCH(ILNGTH)                                                     TES01260
      IF(CARDP(1).EQ.ALPHA(25)) GO TO 33                                      TES01270
      IF(CARDP(1).EQ.ALPHA(14)) GO TO 313                                     TES01280
  315 WRITE(6,106)                                                            TES01290
      GO TO 312                                                               TES01300
  313 WRITE(6,105)                                                            TES01310
  105 FORMAT('0',10X,' IF YOU DESIRE TO HAVE A SMOOTH COPY OF YOUR'/'         TES01320
     * PROGRAM WITH ITS EXECUTION THEN REPLY :YES  ; OTHERWISE    '/'         TES01330
     * REPLY : NO '//)                                                       TES01340
   20  READ(5,101,END=326)CARD                                                TES01350
      CALL CRUNCH(ILNGTH)                                                     TES01360
      IF(CARDP(1).NE.ALPHA(25))GO TO 25                                       TES01370
C                                                                             TES01380
CC    PRINT SMOOTH COPY OF PROGRAM AND EXECUTE IT                             TES01390
C                                                                             TES01400
  314 WRITE(6,104)                                                            TES01410
  104 FORMAT('0'//////////)                                                   TES01420
      DO 13 I=1,NSTMT                                                         TES01430
   13 WRITE(6,103)(SFILE(I,J),J=1,80)                                         TES01440
  103 FORMAT(' ',5X,80A1)                                                     TES01450
  135 CALL INITAL                                                             TES01460
      DEBUG=0.0                                                               TES01470
  136 INBIG=1                                                                 TES01480
      IARRAY=808                                                              TES01490
      NFOR=0                                                                  TES01500
      INTERP=0                                                                TES01510
      IEXERR=0                                                                TES01520
      DO 15 K=1,NSTMT                                                         TES01530
      DO 14 L=1,80                                                            TES01540
   14 CARD(L)=SFILE(K,L)                                                      TES01550
      CALL CRUNCH(ILNGTH)                                                     TES01560
   15 CALL COMPLR(NFOR,IARRAY,ILNGTH,INBIG)                                   TES01570
C                                                                             TES01580
CC    EXECUTION ERROR GO TO EDIT ROUTINE                                      TES01590
C                                                                             TES01600
      IF(IEXERR.EQ.1)GO TO 31                                                 TES01610
      WRITE(6,104)                                                            TES01620
      GO TO 26                                                                TES01630
```

```
   25  IF(CARDP(1).EQ.ALPHA(14))GO TO 26                                    TES01640
   326 WRITE(6,106)                                                         TES01650
   106 FORMAT('0',' *** YOUR REPLY IS INCORRECTLY TYPED REPLY AGAIN***'//TES01660
      * )                                                                  TES01670
       GO TO 20                                                           TES01680
   26  WRITE(6,107)                                                       TES01690
   107  FORMAT('1',10X,' DO YOU WANT TO EXECUTE ANOTHER PROGRAM'/' REPLY
      *: YES  OR  NO .'//)                                               TES01710
   29  READ(5,101,END=330) CARD                                           TES01720
       CALL CRUNCH(ILNGTH)                                                TES01730
       IF(CARDP(1).EQ.ALPHA(14))GO TO 30                                  TES01740
       IF(CARDP(1).EQ.ALPHA(25))GO TO 3                                   TES01750
   330 WRITE(6,106)                                                       TES01760
       GO TO 29                                                           TES01770
   30  RETURN                                                             TES01780
C                                                                         TES01790
CC     EDIT ROUTINE.                                                      TES01800
C                                                                         TES01810
    31 WRITE(6,108)                                                       TES01820
   108 FORMAT('0',10X,' **** EXECUTION ERROR ****                  '//'  TES01830
      *IF YOU WANT TO CORRECT YOUR PROGRAM NOW THEN REPLY : YES  AND '/'  TES01840
      *YOU WILL ENTER THE EDIT MODE TO CORRECT YOUR ERRORS          '/'  TES01850
      *CTHERWISE REPLY : NO  .'//)                                       TES01860
   208 READ(5,101,END=343)CARD                                            TES01870
       CALL CRUNCH(ILNGTH)                                                TES01880
       IF(CARDP(1).NE.ALPHA(25)) GO TO 43                                 TES01890
    33 IF(OLD.EQ.1)GO TO 321                                              TES01900
       WRITE(6,110)                                                       TES01910
   110 FORMAT('0',10X,' **** CAIBASIC EDIT MODE ****               '//'  TES01920
      *BY USING THE REFERENCE NUMBERS LISTED TO THE LEFT OF YOUR BASIC'/'TES01930
      * PROGRAM STATEMENTS YOU MAY ADD , DELETE , OR CORRECT ONE LINE'/'  TES01940
      *OF THE PROGRAM AT A TIME . IN ALL EDITING THE FIRST STEP IS TO'/'  TES01950
      *INPUT THE PROGRAM STATEMENT REFERENCE NUMBER AND HIT THE CARRIAGE'TES01960
      */' RETURN . THE SECOND STEP DEPENDS ON WHAT EDITING YOU DO : '//'  TES01970
      *    1. DELETE                                                 '/'  TES01980
      *THE BASIC STATEMENT REFERENCED IS DELETED BY TYPING THE       '/'  TES01990
      *LETTERS  DEL                                                  '/'  TES02000
      *    2. CORRECT                                                '/'  TES02010
      *TO CORRECT THE BASIC STATEMENT REFERENCED TYPE IN THE COMPLETE'/'  TES02020
      *CORRECT BASIC STATEMENT .                                     '/'  TES02030
      *    3. ADD                                                    '/'  TES02040
      *A BASIC STATEMENT IS ADDED ''AFTER'' THE BASIC STATEMENT      '/'  TES02050
      *REFERENCED BY TYPING THE LETTERS ADD1 FOLLOWED BY THE BASIC   '/'  TES02060
      *STATEMENT . ALL BLANKS FOLLOWING THE LETTERS  ADD1 WILL BE    '/'  TES02070
      *INCLUDED IN THE BASIC STATEMENT . TO PLACE A STATEMENT ''BEFORE'''TES02080
      */' THE FIRST STATEMENT IN THE PROGRAM , USE THE REFERENCE NUMBER OTES02090
      * .'///)                                                           TES02100
       OLD=1                                                            TES02110
```

```
C                                                                       TES02120
CC     WRITE PROGRAM WITH REFERENCE NUMBERS                             TES02130
C                                                                       TES02140
  321 DO 32 I=1,NSTMT                                                   TES02150
   32 WRITE(6,109) I,(SFILE(I,J),J=1,80)                                TES02160
  109 FORMAT(' ',I4,10X,80A1)                                           TES02170
   34 WRITE(6,111)                                                      TES02180
  111 FORMAT('0',' INPUT REFERENCE NUMBER NOW')                         TES02190
   35 KNT=0                                                             TES02200
      READ(5,101,END=331)CARD                                           TES02210
      CALL CRUNCH(ILNGTH)                                               TES02220
      IF(CARDP(1).EQ.ASTRSK) GO TO 35                                   TES02230
      DO 237 I=1,2                                                      TES02240
      DO 235 J=1,10                                                     TES02250
C                                                                       TES02260
CC     CHECK FOR DIGIT                                                  TES02270
C                                                                       TES02280
      IF(CARDP(I).EQ.DIGIT(J)) GO TO 236                                TES02290
  235 CONTINUE                                                          TES02300
      IF(J.EQ.10.AND.CARDP(I).EQ.BLANK) GO TO 237                       TES02310
  331 WRITE(6,106)                                                      TES02320
      GO TO 34                                                          TES02330
C                                                                       TES02340
CC     CONVERT ALPHA CHARACTER TO NUMBER                                TES02350
C                                                                       TES02360
  236 KNT=KNT+1                                                         TES02370
      IF(KNT.EQ.1) NREF=J-1                                             TES02380
      IF(KNT.EQ.2) NREF=10*NREF+(J-1)                                   TES02390
  237 CONTINUE                                                          TES02400
C                                                                       TES02410
CC     CHECK FOR LEGAL REFERENCE NUMBER                                 TES02420
C                                                                       TES02430
      IF(NREF.LE.NSTMT.AND.NREF.GE.0) GO TO 36                          TES02440
      WRITE(6,106)                                                      TES02450
      GO TO 35                                                          TES02460
   36 WRITE(6,112)                                                      TES02470
  112 FORMAT('0',' INPUT EDITING NOW (DEL,BASIC STATEMENT,ADD1....)'//) TES02480
   37 READ(5,101,END=36)CARD                                            TES02490
      CALL CRUNCH(ILNGTH)                                               TES02500
      IF(CARDP(1).EQ.ASTRSK) GO TO 36                                   TES02510
C                                                                       TES02520
CC     DELETE CURRENT LINE                                              TES02530
C                                                                       TES02540
      IF(CARDP(1).EQ.CMINUS) GO TO 50                                   TES02550
C                                                                       TES02560
CC     ADD A STATEMENT AFTER CURRENT LINE                               TES02570
C                                                                       TES02580
      IF(CARDP(1).EQ.PLUS) GO TO 55                                     TES02590
```

```
C                                                                       TES02600
C     CORRECT CURRENT LINE                                              TES02610
C                                                                       TES02620
      DEBUG=0.0                                                         TES02630
      INTERP=1                                                          TES02640
      IEXERR=0                                                          TES02650
      DO 370 I=1,80                                                     TES02660
      IF(CARDP(I).EQ.ALPHA(14).AND.CARDP(I+3).EQ.ALPHA(20)) GO TO 39    TES02670
  370 CONTINUE                                                          TES02680
C                                                                       TES02690
C     IF NO ERRORS , FILE INPUT STATEMENT                               TES02700
C                                                                       TES02710
      CALL COMPLR(NFOR,IARRAY,ILNGTH,INBIG)                             TES02720
      IF(NERRS.EQ.0) GO TO 39                                           TES02730
      NERRS=0                                                           TES02740
      GO TO 34                                                          TES02750
   39 DO 40 J=1,80                                                      TES02760
   40 SFILE(NREF,J)=CARD(J)                                             TES02770
      GO TO 47                                                          TES02780
   43 IF(CARDP(1).EQ.ALPHA(14)) GO TO 44                                TES02790
  343 WRITE(6,106)                                                      TES02800
      GO TO 208                                                         TES02810
   44 WRITE(6,113)                                                      TES02820
  113 FORMAT('0',10X,' IF YOU DESIRE TO HAVE THE PROGRAM RUN WITH    '/'TES02830
     *THE DEBUG FEATURE REPLY: YES  ; OTHERWISE REPLY: NO  AND THIS '/' TES02840
     *PROGRAM WILL BE LOST .'//)                                        TES02850
      CALL CRUNCH(ILNGTH)                                               TES02860
   45 READ(5,101,END=346)CARD                                           TES02870
      IF(CARDP(1).NE.ALPHA(25)) GO TO 46                                TES02880
C                                                                       TES02890
C     EXECUTE WITH DEBUG FEATURE ADDED                                  TES02900
C                                                                       TES02910
      CALL INITAL                                                       TES02920
      DEBUG=1.0                                                         TES02930
      GO TO 136                                                         TES02940
   46 IF(CARDP(1).EQ.ALPHA(14)) GO TO 26                                TES02950
  346 WRITE(6,106)                                                      TES02960
      GO TO 45                                                          TES02970
   47 WRITE(6,114)                                                      TES02980
  114 FORMAT('0',10X,' MORE CORRECTIONS TO BE MADE ??  REPLY: YES  ;'/'  TES02990
     *OR REPLY: NO  AND THE EDITED PROGRAM WILL BE EXECUTED .'//)       TES03000
   48 READ(5,101,END=348)CARD                                           TES03010
      CALL CRUNCH(ILNGTH)                                               TES03020
      IF (CARDP(1).EQ. ALPHA(25)) GO TO 321                             TES03030
      IF (CARDP(1).EQ. ALPHA(14)) GO TO 314                             TES03040
  348 WRITE(6,106)                                                      TES03050
      GO TO 48                                                          TES03060
C                                                                       TES03070
```

```
C      DELETE STATEMENT                                                          TES03080
C                                                                                TES03090
   50 NSTMT=NSTMT-1                                                              TES03100
      DO 52 I=NREF,NSTMT                                                         TES03110
      DO 52 J=1,80                                                               TES03120
   52 SFILE(I,J)=SFILE(I+1,J)                                                    TES03130
      GO TO 47                                                                   TES03140
C                                                                                TES03150
C      ADD A BASIC STATEMENT AFTER CURRENT LINE                                  TES03160
C                                                                                TES03170
   55 IF(NSTMT.GT.99) GO TO 65                                                   TES03180
      NSTMT=NSTMT +1                                                             TES03190
      ITEMP=NSTMT                                                                TES03200
      DEBUG=0.0                                                                  TES03210
      INTERP=1                                                                   TES03220
      IEXERR=0                                                                   TES03230
C                                                                                TES03240
C      FIND BEGINNING OF STATEMENT                                               TES03250
C                                                                                TES03260
      DO 56 K=1,80                                                               TES03270
      IF(CARD(K).EQ.BLANK) GO TO 56                                              TES03280
      IF(CARD(K).NE.ALPHA(1).OR.CARD(K+2).NE.ALPHA(4)) GO TO 56                  TES03290
      IF(CARD(K+3).EQ.DIGIT(2))GO TO 60                                          TES03300
   56 CONTINUE                                                                   TES03310
      WRITE(6,106)                                                               TES03320
      GO TO 37                                                                   TES03330
   60 IBEG=K+3                                                                   TES03340
      DO 600 L=1,80                                                              TES03350
      IBEG=IBEG+1                                                                TES03360
      IF(IBEG.GT.80) GO TO 601                                                   TES03370
      CARD(L)=CARD(IBEG)                                                         TES03380
  600 CONTINUE                                                                   TES03390
  601 CALL CRUNCH(ILNGTH)                                                        TES03400
      IF(CARDP(1).EQ. ASTRSK) GO TO 36                                           TES03410
      DO 603 I=1,80                                                              TES03420
      IF(CARDP(I).EQ.ALPHA(14).AND.CARDP(I+3).EQ.ALPHA(20)) GO TO 602            TES03430
  603 CCNTINUE                                                                   TES03440
      CALL COMPLR(NFOR,IARRAY,ILNGTH,INBIG)                                      TES03450
C                                                                                TES03460
C      IF NO ERRORS THEN FILE STATEMENT                                          TES03470
C                                                                                TES03480
      IF(NERRS.EQ.0) GO TO 602                                                   TES03490
      NERRS=0                                                                    TES03500
      GO TO 36                                                                   TES03510
C                                                                                TES03520
C      ADD STATEMENT TO PROGRAM FILE. MOVE STATEMENTS UP AND DOWN                TES03530
C                                                                                TES03540
  602 IADD=NREF+1                                                                TES03550
```

```
   61 DO 62 I=1,80                                                          TES03560
   62 SFILE(ITEMP,I)=SFILE(ITEMP-1,I)                                       TES03570
      ITEMP=ITEMP-1                                                         TES03580
      IF(ITEMP.GT.IADD) GO TO 61                                            TES03590
      DO 63 J=1,80                                                          TES03600
   63 SFILE(IADD,J)=CARD(J)                                                 TES03610
      GO TO 47                                                              TES03620
   65 WRITE(6,512)                                                          TES03630
      GO TO 47                                                              TES03640
      END                                                                   TES03650

      SUBROUTINE CRUNCH(ILNGTH)                                             CRU00010
C                                                                           CRU00020
C     GIVEN THE VECTOR CARD ; THIS SUBROUTINE REMOVES ALL BLANKS IN CARDP   CRU00030
C     AND RETURNS THE BLANK-LESS VERSION IN CARDP.                          CRU00040
C     THIS ROUTINE ALSO CHECKS FOR TYPING ERRORS  AND CHECKS IF THE USER    CRU00050
C     IS COMPLETED WITH HIS SESSION OR IN AN EDIT MODE                      CRU00060
C                                                                           CRU00070
                                  COMMON                                    CRU00080
     -          STACK(100), PROG(2000), CARD(80), CARDP(80), ALPHA(48),     CRU00090
     -          IAPTR, INPTR, IADATA(500) , XNDATA(500) , STRING(5),        CRU00100
     -          DIGIT(10), IPRITB(10), LISTST(100), ISTLST(100),            CRU00110
     -          PRT(2500), NERRS, INST, NSTLST, DEBUG, DOLSGN, QUOTE,       CRU00120
     -          EQUALS, PARRT, DECMAL, PLUS, CMINUS, SLASH, COMMA,          CRU00130
     -          PARLFT, ASTRSK, BLANK                                       CRU00140
      COMMON  INTERP,IEXERR                                                 CRU00150
      DO 19 I=1,80                                                          CRU00160
   19 CARDP(I)=BLANK                                                        CRU00170
C                                                                           CRU00180
C     REMOVE BLANKS                                                         CRU00190
C                                                                           CRU00200
      ILNGTH=0                                                              CRU00210
      DO 20 I=1,80                                                          CRU00220
      IF (CARD(I).EQ.BLANK) GO TO 20                                        CRU00230
      ILNGTH=ILNGTH+1                                                       CRU00240
      CARDP(ILNGTH)=CARD(I)                                                 CRU00250
   20 CONTINUE                                                              CRU00260
C                                                                           CRU00270
C     CHECK FOR BLANK INPUT                                                 CRU00280
C                                                                           CRU00290
      IF(ILNGTH.EQ.0) CARDP(1)=ASTRSK                                       CRU00300
      DO 25 I=1,80                                                          CRU00310
C                                                                           CRU00320
C     CHECK FOR TYPING ERROR($$$$)                                          CRU00330
C                                                                           CRU00340
      IF(CARDP(I).NE.DOLSGN.OR.CARDP(I+1).NE.DOLSGN) GO TO 26               CRU00350
      IF(CARDP(I+2).EQ.DOLSGN.AND.CARDP(I+3).EQ.DOLSGN) CARDP(1)=ASTRSK     CRU00360
```

```
C                                                                       CRU00370
CC     CHECK FOR END OF SESSION (QUIT)                                  CRU00380
C                                                                       CRU00390
  26   IF(CARDP(I).NE.ALPHA(17).OR.CARDP(I+1).NE.ALPHA(21)) GO TO 27    CRU00400
       IF(CARDP(I+2).EQ.ALPHA( 9).AND.CARDP(I+3).EQ.ALPHA(20)) STOP     CRU00410
C                                                                       CRU00420
CC     CHECK FOR EDIT COMMAND (ADD1)                                    CRU00430
C                                                                       CRU00440
   27  IF(CARDP(I).NE.ALPHA(1).OR.CARDP(I+1).NE.ALPHA(4))GO TO 28       CRU00450
       IF(CARDP(I+2).EQ.ALPHA(4).AND.CARDP(I+3).EQ.DIGIT(2))CARDP(1)=PLUSCRU00460
C                                                                       CRU00470
CC     CHECK FOR EDIT COMMAND (DEL)                                     CRU00480
C                                                                       CRU00490
   28  IF(CARDP(I).NE.ALPHA(4).OR.CARDP(I+1).NE.ALPHA(5)) GO TO 25      CRU00500
       IF(CARDP(I+2).EQ.ALPHA(12))CARDP(1)=CMINUS                       CRU00510
  25   CONTINUE                                                         CRU00520
       RETURN                                                           CRU00530
       END                                                              CRU00540


       SUBROUTINE INITAL
C                                                                       INI00110
CC                                                                      INI00180
CC     THIS ROUTINE INITIALIZES THE SYSTEM. IT LOADS ALL VECTORS        INI00190
CC     WITH APPROPRIATE INFORMATION TO  BE USED BY THE SYSTEM. IT ALSO  INI00200
CC     SETS CERTAIN POINTERS TO THEIR INITIAL VALUES.                   INI00210
C                                                                       INI00020
       COMMON
     -          STACK(100), PROG(2000), CARD(80), CARDP(80), ALPHA(48),  L
     -          IAPTR, INPTR, IADATA(500) , XNDATA(500) , STRING(5),
     -          DIGIT(10), IPRITB(10), LISTST(100), ISTLST(100),
     -          PRT(2500), NERRS, INST, NSTLST, DEBUG, DOLSGN, QUOTE,
     -          EQUALS, PARRT, DECMAL, PLUS, CMINUS, SLASH, COMMA,
     -          PARLFT, ASTRSK, BLANK
       COMMON /RANDNM/IRAND
       DIMENSION ATEMP(26),DIGTMP(10),CHARTM(12)
       DATA ATEMP/1HA,1HB,1HC,1HD,1HE,1HF,1HG,1HH,1HI,1HJ,              INI00130
     - 1HK,1HL,1HM,1HN,1HO,1HP,1HQ,1HR,1HS,1HT,1HU,1HV,1HW,1HX,1HY,1HZ/ INI00140
       DATA DIGTMP/1H0,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9/             INI00150
       DATA CHARTM/1H*,1H ,1H,,1H.,1H=,1H),1H(,1H+,1H',1H$,             INI00160
     - 1H-,1H//                                                         INI00170
C                                                                       INI00220
CC     INITIALIZE                                                       INI00230
C                                                                       INI00240
       IRAND=5555                                                       INI00250
       NERRS=0                                                          INI00260
       INST=0                                                           INI00270
       NSTLST=0                                                         INI00280
```

```
      IAPTR=0                                                          INI00290
      INPTR=0                                                          INI00300
C                                                                      INI00310
C     INITIALIZE PRIORITY TABLE                                        INI00320
      IPRITB(1)=2                                                      INI00330
      IPRITB(2)=2                                                      INI00340
      IPRITB(3)=3                                                      INI00350
      IPRITB(4)=3                                                      INI00360
      IPRITB(5)=4                                                      INI00370
      IPRITB(6)=5                                                      INI00380
C                                                                      INI00390
C     SET UP VOCABULARY                                                INI00400
C     LOAD ALPHABET                                                    INI00410
C                                                                      INI00420
      DO 10 I=1,26                                                     INI00430
   10 ALPHA(I)=ATEMP(I)                                                INI00440
C                                                                      INI00450
C     LOAD DIGITS                                                      INI00460
C                                                                      INI00470
      DO 20 I=1,10                                                     INI00480
   20 DIGIT(I)=DIGTMP(I)                                               INI00490
C                                                                      INI00500
C     LOAD DIGITS INTO ALPHA                                           INI00510
C                                                                      INI00520
      LOC=26                                                           INI00530
      DO 12 I=1,10                                                     INI00540
      LOC=LOC+1                                                        INI00550
   12 ALPHA(LOC)=DIGIT(I)                                              INI00560
C                                                                      INI00570
C     LOAD SPECIAL CHAR INTO ALPHA                                     INI00580
C                                                                      INI00590
      DO 16 I=1,12                                                     INI00600
      LOC=LOC+1                                                        INI00610
   16 ALPHA(LOC)=CHARTM(I)                                             INI00620
C                                                                      INI00630
C     INITIALIZE ARRAY STORAGE POINTERS                                INI00640
C                                                                      INI00650
      DO 50 I=287,384,4
   50 PRT(I)=0.0
C                                                                      INI00680
C     LOAD SPECIAL CHARACTERS                                          INI00690
C                                                                      INI00700
      ASTRSK=CHARTM(1)                                                 INI00710
      BLANK =CHARTM(2)                                                 INI00720
      COMMA =CHARTM(3)                                                 INI00730
      DECMAL=CHARTM(4)                                                 INI00740
      EQUALS=CHARTM(5)                                                 INI00750
      PARRT =CHARTM(6)                                                 INI00760
```

```
      PARLFT=CHARTM(7)                                                  INI00770
      PLUS  =CHARTM(8)                                                  INI00780
      QUOTE =CHARTM(9)                                                  INI00790
      DOLSGN=CHARTM(10)                                                 INI00800
      CMINUS=CHARTM(11)                                                 INI00810
      SLASH =CHARTM(12)                                                 INI00820
C                                                                       INI00830
      RETURN                                                            INI00840
C                                                                       INI00850
      END                                                               INI00860
C                                                                       LES00020
C     THIS LESSON COVERS PROGRAM FORMAT AND MISC INFORMATION SUCH       LES00030
C     AS VARIABLES , NUMBERS , KEY WORDS , EXPRESSIONS , ETC.
C                                                                       LES00050
                                COMMON                                  LES00060
     -          STACK(100), PROG(2000), CARD(80), CARDP(80), ALPHA(48),  LES00070
     -          IAPTR, INPTR, IADATA(500) , XNDATA(500) , STRING(5),     LES00080
     -          DIGIT(10), IPRITB(10), LISTST(100), ISTLST(100),         LES00090
     -          PRT(2500), NERRS, INST, NSTLST, DEBUG, DOLSGN, QUOTE,    LES00100
     -          EQUALS, PARRT, DECMAL, PLUS, CMINUS, SLASH, COMMA,       LES00110
     -          PARLFT, ASTRSK, BLANK                                    LES00120
      COMMON  INTERP,IEXERR                                             LES00130
      REAL*8 DUMY(125)
      REAL*8 LES1/'LESON1  '/
      WRITE(6,100)                                                      LES00140
  100 FORMAT('0',5X,' *** LESSON 1. ***                          '//' LES00150
     *THIS INSTRUCTION SET WILL INTRODUCE YOU TO THE STRUCTURE OF BASIC'LES00160
     */' LANGUAGE STATEMENTS , THE RULES FOR VARIABLES AND NUMBERS , ANDLES00170
     *'/' THE SYMBOLS FOR ARITHMETIC OPERATIONS.'//' ***DONT FORGET TO TLES00180
     *YPE GO AND HIT RETURN WHEN READY TO CONTINUE . ***'//)           LES00190
      READ(5,101,END=301)CARD                                           LES00200
  101 FORMAT(80A1)                                                      LES00210
  301 WRITE(6,102)                                                      LES00220
  102 FORMAT('/',5X,' A.  PROGRAM STRUCTURE                       '/ 'LES00230
     *                  THE BASIC LANGUAGE , AND ALL OTHER LANGUAGES ,'/'LES00240
     * HAS A SPECIFIED STRUCTURE . EACH BASIC STATEMENT HAS A REQUIRED'/LES00250
     *' FORM WITH POSSIBLY ONE OR MORE VARIATIONS. THE FOLLOWING    '/' LES00260
     *EXAMPLE WILL ILLUSTRATE SOME SIMPLE BASIC STATEMENTS IN THE   '/' LES00270
     *PROPER PROGRAM STRUCTURE'//10X,'REM  PROGRAM TO COMPUTE GAS MILAGELES00280
     *'/10X,'REM  M=MILES TRAVELED , G=GAS USED'/10X,' READ M,G'/10X,' LLES00290
     *ET T=M / G '/10X,' PRINT ''MILES TRAVELED'',''GAS USED'','' MILES/LES00300
     *GAL'' '/10X,' PRINT M,G,T '/10X,' DATA 500,25 '/10X,' END'/5X,' OULES00310
     *TPUT PRODUCED IS : '/' MILES TRAVELED  GAS USED    MILES/GAL'/6X,'LES00320
     * 500',11X,'25',10X,'20'//)                                      LES00330
      READ(5,101,END=303)CARD                                           LES00340
  303 WRITE(6,103)                                                      LES00350
  103 FORMAT('0',10X,' AS YOU CAN SEE FROM THE ABOVE SAMPLE, THE PROGRAMLES00360
     * '/' STRUCTURE CONSISTS OF BASIC LANGUAGE STATEMENTS FOLLOWED '/' LES00370
```

```
     *BY AN END STATEMENT . THE WORDS : REM , READ, LET ,  PRINT , DATA'LES00380
     */',AND END   ARE KEY WORDS THAT MAKE UP A BASIC STATEMENT .'//)   LES00390
      READ(5,101,END=304)CARD                                           LES00400
  304 WRITE(6,104)                                                      LES00410
  104 FORMAT('0',10X,' MOST OF THE KEY WORDS USED IN THE BASIC      '/' LES00420
     *STATEMENTS ARE SELF-EXPLANATORY:                              '//5X,'
     *REM  ALLOWS REMARKS/COMMENTS                                  '/5X,'
     *READ M,G  ASSIGNS NUMBERS IN THE DATA STATEMENT TO THE        '/5X,'
     *          VARIABLES M AND G                                   '/5X,'
     *LET  ASSIGNS THE RESULT OF M DIVIDED BY G INTO VARIABLE T     '/5X,'
     *PRINT''STRING''  CAUSES THE STRING IN SINGLE QUOTES TO BE     '/5X,'
     *                 PRINTED LITERALLY                            '/5X,'
     *PRINT M,G,T  PRINTS THE CURRENT VALUE OF THE VARIABLES M,G,T'/5X,'
     * END  TELLS THE COMPUTER THAT THE INPUT PROGRAM IS TO BE      '/5X,'
     *       EXECUTED'//)
      READ(5,101,END=305)CARD                                           LES00500
  305 WRITE(6,105)                                                      LES00510
  105 FORMAT('0',10X,' IF AT THIS POINT YOU WOULD LIKE TO RUN THE    '/' LES00520
     *SAMPLE PROGRAM TO GAIN SOME CONFIDENCE IN THE COMPUTER AND ITS'/' LES00530
     *ABILITY TO PROVIDE SPEEDY RESULTS, THEN REPLY : YES   ; OTHER-'/' LES00540
     *WISE REPLY : NO  AND THE INSTRUCTION WILL CONTINUE .'//)          LES00550
   10 READ(5,101,END=306)CARD                                           LES0056
      CALL CRUNCH(ILNGTH)                                               LES00580
      IF(CARDP(1).EQ.ASTRSK) GO TO 10                                   LES00570
      IF (CARDP(1).EQ.ALPHA(14)) GO TO 25                               LES00590
      IF (CARDP(1).EQ.ALPHA(25)) GO TO 15                               LES00600
  306 WRITE(6,106)                                                      LES00610
  106 FORMAT('0','*** YOUR REPLY WAS TYPED INCORRECTLY ; CHECK THE   '/' LES00620
     *QUESTION AND REPLY AGAIN ***'//)                                  LES00630
      GO TO 10                                                          LES00640
C                                                                       LES00650
CC    EXECUTE SAMPLE PROGRAM                                            LES00660
C                                                                       LES00670
   15 WRITE(6,107)                                                      LES00680
  107 FORMAT('0',10X,' TO EXECUTE THE SAMPLE PROGRAM TYPE IN THE BASIC'/LES00690
     *' STATEMENTS AS THEY APPEAR , AND DONT WORRY ABOUT SPACING .THE'/'LES00700
     * COMPUTER ANALIZES EACH BASIC STATEMENT AS IT IS INPUT ; AND    '/'LES00710
     * IF THERE ARE NO ERRORS , IT UNLOCKS THE KEYBOARD AND WAITS FOR'/'LES00720
     * YOUR NEXT INPUT . WHEN THE END STATEMENT IS INPUT THE PROGRAM'/' LES00730
     *IS EXECUTED .'//)                                                 LES00740
      READ(5,101,END=308)CARD                                           LES00750
  308 WRITE(6,108)                                                      LES00760
  108 FORMAT('0',10X,' HOWEVER , IF YOU MAKE AN ERROR THE COMPUTER WILL'LES00770
     *'/' TELL YOU THE ERROR , AND EXPECT A CORRECTION . DONT WORRY ABOUTLES00780
     *'/' THE ERROR , CHECK YOUR INPUT AGAINST THE SAMPLE PROGRAM , AND'LES00790
     *'/' INPUT  THE CORRECT STATEMENT . IF YOU MAKE A TYPING MISTAKE '/'LES00800
     * , INPUT FOUR ASTERIX(****),HIT RETURN , AND THEN INPUT THE    '/' LES00810
     *CORRECT STATEMENT .'/10X,' YOU MAY USE ANY PAIR OF INTEGER OR DECILES00820
```

```
   *MAL NUMBERS'/' FOR INPUT DATA.YOU MAY OMIT THE REM STATEMENTS AND LES00830
   * THE PRINT ''MILES TRAVELED''.... ETC'/' STATEMENT IF YOU DONT WANLES00840
   *T TO TYPE A LOT OF STATEMENTS'//)                                  LES00850
    CALL TEST1                                                        LES00860
 25 WRITE(6,111)                                                      LES00870
111 FORMAT('0', 5X,'B. PROGRAM FORMAT'// 5X,' A BASIC PROGRAM CONSISTS
   * OF A SEQUENCE OF BASIC STATEMENTS , ONE'/' STATEMENT PER INPUT LILES00890
   *NE , FOLLOWED BY AN END STATEMENT . '/' BECAUSE NO BASIC STATEMENTLES00900
   * MAY BE LONGER THAN ONE INPUT LINE (80 SPACES) ,'/' THERE IS NO PRLES00910
   *OVISION FOR CONTINUING STATEMENTS FROM ONE LINE'/' TO THE NEXT . HLES00920
   *OWEVER ; YOU MAY SPACE THE INPUT LINE '/' AS DESIRED FOR READABILILES00930
   *TY SINCE THE COMPUTER IGNORES BLANKS IN BASIC .'//)                LES00940
    READ(5,101,END=309)CARD                                           LES00950
309 WRITE(6,112)                                                      LES00960
112 FORMAT('0',10X,' 1. STATEMENT NUMBERS                        '/   LES00970
   *              ' EACH BASIC STATEMENT MAY HAVE AN OPTIONAL     '/' LES00980
   *STATEMENT NUMBER PRECEDING IT FOR IDENTIFICATION PURPOSES .    '/' LES00990
   *THIS STATEMENT NUMBER MUST BE AN INTEGER BETWEEN 1 -> 9999      '/LES01000
   *5X,' FOR EXAMPLE :    12 READ M,G                              '// LES01010
   *10X,' 2. KEY WORDS                '/' THE KEY WORDS THAT MAKE UP A LES01020
   *BASIC STATEMENT (REM ,READ ,LET ,ETC,)'/' ARE SPECIAL TERMINAL SYMLES01030
   *BOLS THAT ARE RECOGNIZED BY THE COMPUTER'/' AND FOR THIS REASON THLES01040
   *EY MUST BE SPELLED CORRECTLY AND ONLY'/' USED IN BASIC STATEMENTS LES01050
   *.'/10X,' THE END STATEMENT INDICATES THAT THE INPUT PROGRAM IS'/' LES01060
   *COMPLETED AND THAT PROGRAM EXECUTION IS TO BEGIN . THE ''END'''/' LES01070
   *STATEMENT IS ALWAYS THE LAST STATEMENT IN A PROGRAM .'//)          LES01080
    READ(5,101,END=313)CARD                                           LES01090
313 WRITE(6,113)                                                      LES01100
113 FORMAT('0',' YOU WILL NOW BE ASKED A FEW SIMPLE QUESTIONS ABOUT'/'LES01110
   * WHAT YOU HAVE JUST LEARNED .'//)                                 LES01120
    WRITE(6,114)                                                      LES01130
114 FORMAT('0',10X,' 1.) IS THIS A LEGAL BASIC PROGRAM(REPLY : YES OR LES01140
   *NO)??'//5X,' REM ONE LINE DO NOTHING PROGRAM'/5X,' END'///)       LES01150
 30 READ(5,101,END=315)CARD                                           LES01160
    CALL CRUNCH(ILNGTH)                                               LES01170
    IF(CARDP(1).EQ.ASTRSK) GO TO 30                                   LES01180
    IF(CARDP(1).EQ.ALPHA(14).OR.CARDP(1).EQ.ALPHA(25)) GO TO 31       LES01190
315 WRITE(6,106)                                                      LES01200
    GO TO 30                                                          LES01210
 31 WRITE(6,115)                                                      LES01220
115 FORMAT('0',' YES , THE SIMPLEST BASIC PROGRAM CONSISTS OF JUST AN LES01230
   * ''END'' STATEMENT.')                                            LES01240
    WRITE(6,116)                                                      LES01250
116 FORMAT('0',10X,' 2.) WHICH OF THE FOLLOWING BASIC STATEMENTS IS'/'LES01260
   * IN THE PROPER FORMAT :'/15X,' A.) 15 LET T=M/G'/15X,' B.) 15LETT=LES01270
   *M/G'/15X,' C.) 15  L E T   T = M / G '//10X,' REPLY A , B , C  OR LES01280
   *ALL .'//)                                                         LES01290
 32 READ(5,101,END=317)CARD                                           LES01300
```

```
      CALL CRUNCH(ILNGTH)                                                    LES01310
      IF(CARDP(1).EQ.ASTRSK) GO TO 32                                        LES01320
      IF(CARDP(1).EQ.ALPHA(1).OR.CARDP(1).EQ.ALPHA(2)) GO TO 33              LES01330
      IF(CARDP(1).EQ.ALPHA(3)) GO TO 33                                      LES01340
  317 WRITE(6,106)                                                           LES01350
      GO TO 32                                                               LES01360
   33 WRITE(6,117)                                                           LES01370
  117 FORMAT('0',' ALL OF THE ABOVE BASIC STATEMENTS ARE CORRECT BECAUSE'LES01380
     *'/' SPACES ARE DISREGARDED . NOTE THAT STATEMENT B.) , ALTHOUGH'/'LES01390
     * CORRECT IS CONFUSING TO READ . BLANKS AND INDENTATIONS MAKE A'/' LES01400
     *PROGRAM EASY TO READ .'/)                                           LES01410
      WRITE(6,218)                                                           LES01420
  218 FORMAT('0',5X,' C. ALPHA-NUMERIC CHARACTERS                '// 5X,'
     *ALPHA-NUMERIC CHARACTERS ARE THE LEGAL CHARACTERS , DIGITS , AND'/LES01440
     *' SPECIAL CHARACTERS THAT CAN BE USED IN BASIC .          '/10X,' LES01450
     * 1. CHARACTERS                                                 '/' LES01460
     *CHARACTERS CONSIST OF THE LETTERS IN THE ALPHABET  A-->Z  '//10X, LES01470
     *' 2. DIGITS                                                   '/' LES01480
     *DIGITS ARE THE SINGLE NUMBERS  0-->9                      '//10X,  LES01490
     *' 3. SPECIAL CHARACTERS                                       '/' LES01500
     * THE SPECIAL CHARACTERS ARE  **  *  /  +  -  (  )  =  ''  , $'//10
     *X,' 4. STRING'/' A STRING IS ANY LIST OF ALPHA-NUMERIC CHARACTERS'LES01520
     */' ENCLOSED IN ''SINGLE'' QUOTES . FOR EXAMPLE :'/5X,' ''THIS IS ALES01530
     * STRING'' '//)                                                     LES01540
      READ(5,101,END=318)CARD                                                LES01550
  318 WRITE(6,118)                                                           LES01560
  118 FORMAT('0', 5X,' D. VARIABLES                                 '// LES01570
     *10X,' IN BASIC THERE ARE THREE TYPES OF VARIABLES : SIMPLE , ALPHALES01580
     *'/' AND SUBSCRIPTED . SUBSCRIPTED VARIABLES WILL BE COVERED IN'/' LES01590
     *LESSON 5 .'/10X,' 1. SIMPLE VARIABLES                        '// LES01600
     *                  ' SIMPLE VARIABLES ARE IDENTIFIED BY A SINGLE LETTLES01610
     *ER'/' OR A SINGLE LETTER FOLLOWED BY A DIGIT BETWEEN 0->9 .     '//LES01620
     *5X,' FOR EXAMPLE : A , A3 , Z0 , AND X  ARE LEGAL VARIABLES ; BUT'LES01630
     */20X,' A26 , 9Z , ABC , AND X1Y  ARE ILLEGAL VARIABLES .'/' THEREFLES01640
     *ORE ; YOU HAVE 286 SIMPLE VARIABLES FOR USE IN YOUR PROGRAMS.'//) LES01650
      READ(5,101,END=319)CARD                                                LES01660
  319 WRITE(6,119)                                                           LES01670
  119 FORMAT('0',10X,' 2. ALPHA VARIABLES                          '/   LES01680
     *                  ' ALPHA VARIABLES ARE USED FOR ALPHA-NUMERIC MANIPULES01690
     *LATIONS'/' IN WHICH A GROUP OF ALPHA-NUMERIC CHARACTERS , CALLED'/LES01700
     *' A STRING , ARE REPRESENTATED BY AN ALPHA VARIABLE . THE ALPHA'/'LES01710
     * VARIABLE CONSISTS OF A SINGLE LETTER FOLLOWED BY A DOLLAR SIGN , LES01720
     *$.'/' THE MAXIMUM LENGTH OF THE ALPHA-NUMERIC STRING ASSIGNED'/'  LES01730
     *TO THE ALPHA VARIABLE IS 16 CHARACTERS.'/                         LES01740
     *'          FOR EXAMPLE : A$ , X$ , Z$  ARE LEGAL ALPHA VARIABLES.'/LES01750
     *' A SIMPLE USE OF ALPHA VARIABLES FOLLOWS :'//5X,' READ A$,B$'/ 5XLES01760
     *,' PRINT A$,B$'/5X,' DATA ''MONDAY'',''21 JUNE'''/5X,' END'//' THILES01770
     *S PROGRAM WILL PRODUCE THE OUTPUT :'//10X,' MONDAY 21 JUNE'//)    LES01780
```

```
    READ(5,101,END=320)CARD                                                 LES01790
320 WRITE(6,120)                                                            LES01800
120 FORMAT('0', 5X,' E.NUMBERS                                      '//LES01810
   *10X,' NUMBERS MAY BE EXPRESSED AS INTEGERS OR AS REALS ,AND A '/' LES01820
   *NUMBER WHETHER INTEGER OR REAL IS LIMITED TO 9 DIGITS , NOT INCLUDLES01830
   *ING'/' DECIMAL POINT. A NUMBER IS ASSUMED POSITIVE UNLESS IT  '/' LES01840
   *IS PRECEEDED BY A -  SIGN .                                      '//LES01850
   *10X,' 1. INTEGERS                                  '//10X,'    INTEGELES01860
   *RS ARE NUMBERS WITH NO FRACTIONAL PART ,IE. 3 , 15                '//
   *10X,' 2. REALS                                              '//10X,' LES01880
   * REAL (FIXED-POINT) NUMBERS HAVE A DECIMAL POINT AND A FRACTIONAL'LES01890
   */' PART ,IE. 3.0 , 15.31, 729.1 , 0.0 '//)                          LES01900
    READ(5,101,END=321)CARD                                             LES01910
321 WRITE(6,121)                                                        LES01920
121 FORMAT('0', 5X,' F. EXPRESSIONS                                 '/' LES01930
   *AN EXPRESSION MAY BE A SINGLE CONSTANT OR VARIABLE , OR AN    '/' LES01940
   *ARITHMETIC EXPRESSION . ARITHMETIC EXPRESSIONS ARE FORMED BY  '/' LES01950
   *USING OPERATORS AND PARENTHESIS .'//10X,' 1. OPERATORS        '// LES01960
   *10X,' NUMBERS AND SIMPLE VARIABLES MAY BE COMBINED INTO ARITHMETICLES01970
   *'/' EXPRESSIONS BY USING ONE OR MORE OF THE ARITHMETIC OPERATORS :LES01980
   *'//4X,' **  EXPONENTIATION'/5X,' *  MULTIPLICATION'/5X,' /  DIVISILES01990
   *ON'/5X,' +  ADDITION'/5X,' -  SUBTRACTION                       '//' LES02000
   *IN WRITING AN EXPRESSION , EACH ARITHMETIC OPERATION MUST BE SHOWNLES02010
   *.'/' NO TWO OPERATIONS MAY BE ADJACENT , NOR MAY TWO NUMBERS OR'/'LES02020
   * VARIABLES BE ADJACENT IN AN EXPRESSION.'/' FOR EXAMPLE: 2+2 , A/BLES02030
   *, B+C , X*Z , AND Y-3.0  ARE  LEGAL EXPRESSIONS.'//)                LES02040
    READ(5,101,END=322)CARD                                             LES02050
322 WRITE(6,122)                                                        LES02060
122 FORMAT('0',10X,' 2. PARENTHESIS                                 '//'LES02070
   *                  PARENTHESIS MAY BE USED TO GROUP EXPRESSIONS AND'LES02080
   */' TO CONTROL THE ORDER IN WHICH AN ARITHMETIC EXPRESSION IS EVALULES02090
   *ATED .'/' THE NORMAL HIERARCHY OF OPERATORS IS : '/5X,' 1. **'/5X,LES02100
   *' 2. *  AND  /'/5X,' 3. +  AND  -                              '//' LES02110
   *IF PARENTHESIS ARE USED , THE EXPRESSION WITHIN A PARENTHESIS PAIRLES02120
   *'/' IS EVALUATED FIRST . PARENTHESIS IN EXPRESSIONS MUST ALWAYS'/'LES02130
   * CCCUR IN PAIRS ,IE.  5 * (4 + 3) , (5*(4+3))  ARE EQUIVALENT '/' LES02140
   *EXPRESSIONS HAVING THE VALUE 35 .                                '//'LES02150
   * OPERATIONS ARE EVALUATED ACCORDING TO THE OPERATORS HIERARCHY'/' LES02160
   *IF TWO OPERATORS OF THE SAME HIERARCHY OCCUR IN AN EXPRESSION '/' LES02170
   *EVALUATION IS FROM LEFT TO RIGHT .'/' FOR EXAMPLE: 2*4 + 6/3 '//' LES02180
   *IN THIS EXPRESSION  2*4  WILL BE EVALUATED FIRST , THEN  6/3 ,'/' LES02190
   *AND THEN THE TWO RESULTS WILL BE ADDED .'//)                         LES02200
    READ(5,101,END=323)CARD                                             LES02210
323 WRITE(6,113)                                                        LES02220
    WRITE(6,123)                                                        LES02230
123 FORMAT('0',10X,' 1.) 4 + 6 / 2  HAS THE VALUE (REPLY WITH VALUE)'/LES02240
   */)                                                                  LES02250
 34 READ(5,101,END=324)CARD                                             LES02260
```

```
      CALL CRUNCH(ILNGTH)                                                LES02270
      IF(CARDP(1).EQ.ASTRSK) GO TO 34                                    LES02280
      IF(CARDP(1).EQ.DIGIT(8)) GO TO 37                                  LES02290
      DO 35 I=1,10                                                       LES02300
      IF(CARDP(1).EQ.DIGIT(I)) GO TO 36                                  LES02310
   35 CONTINUE                                                           LES02320
  324 WRITE(6,106)                                                       LES02330
      GO TO 34                                                           LES02340
  124 FORMAT('0',' YOUR ANSWER IS INCORRECT . THE EXPRESSION IS EVALUATELES02350
     *D AS FOLLOWS :')                                                   LES02360
   36 WRITE(6,124)                                                       LES02370
      WRITE(6,125)                                                       LES02380
  125 FORMAT('0',10X,' 4+6/2  ->  4+3  ->  7 ')                          LES02390
   37 WRITE(6,126)                                                       LES02400
  126 FORMAT('0',10X,' 2.) (4+6)/2  HAS THE VALUE (REPLY WITH VALUE)'//)LES02410
   38 READ(5,101,END=327)CARD                                            LES02420
      CALL CRUNCH(ILNGTH)                                                LES02430
      IF(CARDP(1).EQ.ASTRSK) GO TO 38                                    LES02440
      IF(CARDP(1).EQ.DIGIT(6)) GO TO 41                                  LES02450
      DO 39 I=1,10                                                       LES02460
      IF(CARDP(1) .EQ.DIGIT(I)) GO TO 40                                 LES02470
   39 CONTINUE                                                           LES02480
  327 WRITE(6,106)                                                       LES02490
      GO TO 38                                                           LES02500
   40 WRITE(6,124)                                                       LES02510
      WRITE(6,127)                                                       LES02520
  127 FORMAT('0',10X,' (4+6)/2  ->  10/2  ->  5')                        LES02530
   41 WRITE(6,128)                                                       LES02540
  128 FORMAT('0',10X,' 3.) ((4+6)/2)**2  HAS THE VALUE (REPLY : VALUE)'/LES02550
     */)                                                                 LES02560
   42 READ(5,101,END=46)CARD                                             LES02570
      CALL CRUNCH(ILNGTH)                                                LES02580
      IF(CARDP(1).EQ.ASTRSK) GO TO 42                                    LES02590
      IF(CARDP(1).EQ.DIGIT(3).AND.CARDP(2).EQ.DIGIT(6))  GO TO 48        LES02600
      DO 43 I=1,10                                                       LES02610
      IF(CARDP(1).EQ.DIGIT(I)) GO TO 44                                  LES02620
   43 CONTINUE                                                           LES02630
      GO TO 46                                                           LES02640
   44 CONTINUE                                                           LES02650
      DO 45 J=1,10                                                       LES02660
      IF(CARDP(2).EQ.DIGIT(J)) GO TO 47                                  LES02670
   45 CONTINUE                                                           LES02680
   46 WRITE(6,106)                                                       LES02690
      GO TO 42                                                           LES02700
   47 WRITE(6,124)                                                       LES02710
      WRITE(6,129)                                                       LES02720
  129 FORMAT('0',10X,' ((4+6)/2)**2  ->  (10/2)**2  ->  (5)**2  -> 25'/)LES02730
   48 WRITE(6,130)                                                       LES02740
```

```
130 FORMAT('0', 5X,'  F. SUMMARY                                          '/' LES02750
   *                THIS CONCLUDES THE INSTRUCTION SET FOR LESSON 1.'LES02760
   */' YOU WILL NOW BE GIVEN A SAMPLE BASIC PROGRAM AND ASKED TO FIND'LES02770
   */' THE MISTAKES IN IT , CONCERNING WHAT YOU HAVE LEARNED IN THIS 'LES02780
   */' LESSON :'//5X,' 1  REM REVIEW PROGRAM '/5X,' 2  READ A1,B2,Z1,$LES02790
   *Y'/5X,' 3  LET A1=B2(A1+Z1)'/6X,'4  LET B2=((A1*3)+Z1**2)'/5X,' 5
   *  LET Z1=3.141592763'/5X,' 6  PRINT 1A,B2,Z1'/5X,' 7  DATA 5.0,3,3.LES02810
   *1416,''FINI'''//' AFTER LOOKING AT THE SAMPLE PROBLEM , YOU WILL BLES02820
   *E ASKED QUESTIONS'/' ABOUT EACH STATEMENT ,'//)                      LES02830
    READ(5,101,END=331)CARD                                             LES02840
331 WRITE(6,131)                                                        LES02850
131 FORMAT('0',10X,' 1.)  LINE 2  CONTAINS A READ STATEMENT FOLLOWED'/LES02860
   *' BY A LIST OF VARIABLES . ANY ERRORS (REPLY: YES OR NO)?'//)       LES02870
 50 READ(5,101,END=332)CARD                                             LES02880
    CALL CRUNCH(ILNGTH)                                                 LES02890
    IF(CARDP(1).EQ.ASTRSK) GO TO 50                                     LES02900
    IF(CARDP(1).EQ.ALPHA(14)) GO TO 52                                  LES02910
    IF(CARDP(1).EQ.ALPHA(25)) GO TO 51                                  LES02920
332 WRITE(6,106)                                                        LES02930
    GO TO 50                                                            LES02940
 51 WRITE(6,132)                                                        LES02950
132 FORMAT('0',' INPUT CORRECTION TO THE INCORRECT VARIABLE ONLY .'/)    LES02960
510 READ(5,101,END=52)CARD                                              LES02970
    CALL CRUNCH(ILNGTH)                                                 LES02980
    IF(CARDP(1).EQ.ASTRSK) GO TO 510                                    LES02990
    IF(CARDP(1).EQ.ALPHA(25).AND.CARDP(2).EQ.ALPHA(46)) GO TO 53        LES03000
 52 WRITE(6,133)                                                        LES03010
133 FORMAT('0',' THE ILLEGAL VARIABLE IS $Y . SIMPLE VARIABLES ARE  '/LES03020
   *' A LETTER OR A LETTER FOLLOWED BY A SINGLE DIGIT ; ALPHA VARIABLELES03030
   *S'/' ARE A LETTER FOLLOWED BY A DOLLAR SIGN , $, IE. Y$ .')         LES03040
53  WRITE(6,144)                                                       LES03050
144 FORMAT('0',10X,' 2.)  LINE 3  CONTAINS A LET STATEMENT FOLLOWED'/'LES03060
   * BY A1 = EXPRESSION . ANY ERRORS (REPLY : YES OR NO) ?'//)         LES03070
54  READ(5,101,END=345)CARD                                            LES03080
    CALL CRUNCH(ILNGTH)                                                 LES03090
    IF(CARDP(1).EQ.ASTRSK) GO TO 54                                     LES03100
    IF(CARDP(1).EQ.ALPHA(14))GO TO 58                                   LES03110
    IF(CARDP(1).EQ.ALPHA(25))GO TO 55                                   LES03120
345 WRITE(6,106)                                                        LES03130
    GO TO 54                                                            LES03140
 55 WRITE(6,134)                                                        LES03150
134 FORMAT('0',' INPUT CORRECTION TO ILLEGAL EXPRESSION ; EVERYTHING'/LES03160
   *' AFTER = SIGN .'//)                                                LES03170
 56 READ(5,101,END=58)CARD                                              LES03180
    CALL CRUNCH(ILNGTH)                                                 LES03190
    IF(CARDP(1).EQ.ASTRSK) GO TO 56                                     LES03200
    IF(CARDP(1).EQ.ALPHA(2).AND.CARDP(2).EQ.DIGIT(3)) GO TO 57          LES03210
 57 IF(CARDP(3).NE.ASTRSK .OR.CARDP(4).NE.PARLFT)GO TO 58               LES03220
```

```
      IF(CARDP(5).NE.ALPHA(1) .OR.CARDP(6).NE.DIGIT(2))GO TO 58          LES03230
      IF(CARDP(7).NE.PLUS .OR.CARDP(8).NE.ALPHA(26))GO TO 58             LES03240
      IF(CARDP(9).EQ.DIGIT(2).AND.CARDP(10).EQ.PARRT)GO TO 59            LES03250
   58 WRITE(6,135)                                                       LES03260
  135 FORMAT('0',' THE EXPRESSION SHOULD BE '/10X,' A1=B2*(A1+Z1)   '/' LES03270
     *EACH ARITHMETIC OPERATION MUST BE WRITTEN OUT ; A(B)  IS NOT ASSUMLES03280
     *ED'/' TO BE A*(B) .')                                             LES03290
   59 WRITE(6,136)                                                       LES03300
  136 FORMAT('0',10X,' 3.)  LINE 4  IS SIMILAR TO LINE 3 . ANY ERRORS'/'LES03310
     * REPLY   YES OR NO) ?'//)                                         LES03320
   60 READ(5,101,END=361)CARD                                            LES03330
      CALL CRUNCH(ILNGTH)                                               LES03340
      IF(CARDP(1).EQ.ASTRSK) GO TO 60                                   LES03350
      IF(CARDP(1).EQ.ALPHA(25))GO TO 64
      IF(CARDP(1).EQ.ALPHA(14)) GO TO 65
  361 WRITE(6,106)                                                      LES03380
      GO TO 60                                                          LES03390
   64 WRITE(6,137)                                                      LES03510
  137 FORMAT('0',' THE EXPRESSION IS CORRECT .')
   65 WRITE(6,138)                                                      LES03540
  138 FORMAT('0',10X,' 4.)  LINE 5  CONTAINS A LET STATEMENT FOLLOWED'/'LES03550
     * BY A NUMBER . ANY ERRORS (REPLY : YES OR NO) ?'//)               LES03560
   70  READ(5,101,END=371)CARD                                          LES03570
      CALL CRUNCH(ILNGTH)                                               LES03580
      IF(CARDP(1).EQ.ASTRSK) GO TO 70                                   LES03590
      IF(CARDP(1).EQ.ALPHA(14).OR.CARDP(1).EQ.ALPHA(25)) GO TO 71       LES03600
  371 WRITE(6,106)                                                      LES03610
      GO TO 70                                                          LES03620
   71 WRITE(6,139)                                                      LES03630
  139 FORMAT('0',' THE NUMBER CONTAINS 10 DIGITS AND THE MAXIMUM ALLOWEDLES03640
     *'/' IS 9 DIGITS.')                                               LES03650
      WRITE(6,140)                                                      LES03660
  140 FORMAT('0',10X,' 5.)  LINE 6  CONTAINS A PRINT STATEMENT FOLLOWED'LES03670
     */' BY A LIST OF VARIABLES . ANY ERRORS (REPLY : YES OR NO)?'//)   LES03680
   72 READ(5,101,END=373)CARD                                           LES03690
      CALL CRUNCH(ILNGTH)                                               LES03700
      IF(CARDP(1).EQ.ASTRSK) GO TO 72                                   LES03710
      IF(CARDP(1) .EQ.ALPHA(14)) GO TO 74                               LES03720
      IF(CARDP(1).EQ.ALPHA(25)) GO TO 73                                LES03730
  373 WRITE(6,106)                                                      LES03740
      GO TO 72                                                          LES03750
   73 WRITE(6,132)                                                      LES03760
  730 READ(5,101,END=74)CARD                                            LES03770
      CALL CRUNCH(ILNGTH)                                               LES03780
      IF(CARDP(1).EQ.ASTRSK) GO TO 730                                  LES03790
      IF(CARDP(1).EQ.ALPHA(1).AND.CARDP(2).EQ.DIGIT(2)) GO TO 75        LES03800
   74 WRITE(6,141)                                                      LES03810
  141 FORMAT('0',' THE ILLEGAL VARIABLE IS 1A . SIMPLE VARIABLES ARE A'/LES03820
```

```
     *' LETTER OR A LETTER FOLLOWED BY A SINGLE DIGIT ,IE. A1 .')        LES03830
  75 WRITE(6,142)                                                       LES03840
 142 FORMAT('0',10X,' THERE ARE NO ERRORS IN LINE 7 . IS THE PROGRAM'/'LES03850
     * READY TO EXECUTE (ASSUMING ABOVE ERRORS CORRECTED)'/' (REPLY : YE
     *S OR NO)'//)                                                      LES03870
 750 READ(5,101,END=80)CARD                                             LES03880
     CALL CRUNCH(ILNGTH)                                                LES03890
     IF(CARDP(1).EQ.ASTRSK) GO TO 750                                   LES03900
     IF(CARDP(1).EQ.ALPHA(14).OR.CARDP(1).EQ.ALPHA(25)) GO TO 80        LES03910
     WRITE(6,106)                                                       LES03920
  80 WRITE(6,143)                                                       LES03930
 143 FORMAT('0',10X,' THE PROGRAM FORMAT REQUIRES THAT AN END STATEMENTLES03940
     *'/' BE THE LAST STATEMENT OF THE PROGRAM . THUS THIS SAMPLE PROGRALES03950
     *M'/' WOULD NOT EXECUTE .'//' THIS CONCLUDES THE REVIEW QUESTIONS'/LES03960
     *' FROM LESSCN 1 .'//)                                             LES03970
      CALL EXIT
      END
C                                                                       LES00020
CC    LESSON 2 INTRODUCES THE  REM , PRINT , READ , DATA , RESTORE ,    LES00030
CC    AND RESTORE$ STATEMENTS                                           LES00040
C                                                                       LES00050
                                COMMON                                  LES00060
     -          STACK(100), PROG(2000), CARD(80), CARDP(80), ALPHA(48),  LES00070
     -          IAPTR, INPTR, IADATA(500) , XNDATA(500) , STRING(5),     LES00080
     -          DIGIT(10), IPRITB(10), LISTST(100), ISTLST(100),         LES00090
     -          PRT(2500), NERRS, INST, NSTLST, DEBUG, DOLSGN, QUOTE,    LES00100
     -          EQUALS, PARRT, DECMAL, PLUS, CMINUS, SLASH, COMMA,       LES00110
     -          PARLFT, ASTRSK, BLANK                                    LES00120
      COMMCN  INTERP,IEXERR                                             LES00130
      REAL*8 LES2/'LESCN2  '/
      CALL ALOAD(LES2,N1)
      WRITE(6,100)                                                      LES00140
 100 FORMAT('0',5X,' *** LESSON 2. ***                              '//' LES00150
     *THIS INSTRUCTION SEQUENCE WILL INTRODUCE YOU TO THE BASIC LANGUAGELES00160
     *'/' STATEMENTS  REM , PRINT , READ , AND DATA . USING THESE STATEMLES00170
     *ENTS'/' YOU WILL BE ABLE TO CONSTRUCT AND EXECUTE ELEMENTARY PROGRLES00180
     *AMS .'/ 1X,' THE FORM FOR EACH BASIC STATEMENT WILL INCLUDE :'//10
     *X,'''KEY WORD'' << ELEMENTS OR LIST OF ELEMENTS SEPARATED BY COMMALES00200
     *S >>                                                          '//' LES00210
     *THE CARET SYMBOLS ''<< >>'' DELINIATE THE LEGAL ITEMS THAT MAY'/' LES00220
     *FOLLOW THE KEYWORD AND MAKE UP THE BASIC STATEMENT .'//)           LES00230
      READ(5,101,END=302)CARD                                           LES00240
 101 FORMAT(80A1)                                                       LES00250
 302 WRITE(6,102)                                                       LES00260
 102 FORMAT('0',10X,' A. REM                                            LES00270
     *          '/5X,' THE BASIC STATEMENT WHICH ALLOWS YOU TO INSERT '/LES00280
     *' REMARKS INTO YOUR PROGRAM IS IDENTIFIED BY THE KEY WORD ''REM''.LES00290
     *'/' REM IS A NON-EXECUTING STATEMENT WHICH MAY BE USED OPTIONALLY'LES00300
```

```
   */' AT ANY PLACE IN YOUR PROGRAM TO INTRODUCE A PROGRAM NAME , TO'/LES00310
   *' EXPLAIN VARIABLES, TO DOCUMENT YOUR PROGRAM , ETC.'//)          LES00320
    READ(5,101,END=303)CARD                                          LES00330
303 WRITE(6,103)                                                     LES00340
103 FORMAT('0',5X, ' THE FORM FOR THE REM STATEMENT IS :     '//15X,' LES00350
   * REM << ANY SEQUENCE OF ALPHA-NUMERIC CHARACTERS >>          '//' LES00360
   *THE REM STATEMENT IS IGNORED BY THE CAIBASIC COMPILER , AND IS'/' LES00370
   *CNLY FOR YOUR INFORMATION .'15X,' FOR EXAMPLE :'//10X,' REM  PROGRLES00380
   *AM TO COMPUTE INCOME TAX'//)                                     LES00390
    READ(5,101,END=304)CARD                                          LES00400
304 WRITE(6,104)                                                     LES00410
104 FORMAT('0',10X,' B. PRINT                                '/5X,' LES00420
   *THE PRINT STATEMENT IS THE METHOD OF WRITING OUT THE RESULTS OF'/'LES00430
   * THE BASIC PROGRAM , TO DISPLAY VALUES OF VARIABLES , TO LABEL'/' LES00440
   *THE RESULTS , AND TO SKIP A LINE. THE FORM OF THE PRINT      '/'
   *STATEMENT IS :'//1X,' PRINT << EXPRESSION OR ''STRING'',....,EXPRE
   *SSION OR ''STRING'' >>'//' MULTIPLE ELEMENTS IN THE PRINT LIST ARELES00470
   * SEPARARED BY COMMAS.'//)                                        LES00480
    READ(5,101,END=305)CARD                                          LES00490
305 WRITE(6,105)                                                     LES00500
105 FORMAT('0', 5X,' 1. PRINT << EXPRESSION >>  WILL WRITE OUT THE'/'
   *CURRENT VALUE OF THE EXPRESSION , WHERE AN EXPRESSION AS DEFINED'/LES00520
   *' IN LESSON 1 WAS A NUMBER , A VARIABLE , OR AN ARITHEMTIC    '/' LES00530
   *EXPRESSION THAT IS TO BE EVALUATED .'/10X,' FOR EXAMPLE : '//5X,' LES00540
   *PRINT 1,A,B1,5**2'//' ASSUMING THAT A=10.0 , B=13.3  WOULD PRINT :LES00550
   *'//5X,' 1          10.0          13.3          25'//)             LES00560
    READ(5,101,END=306)CARD                                          LES00570
306 WRITE(6,106)                                                     LES00580
106 FORMAT('0', 5X,' 2. PRINT <<''STRING'' >>  WILL PRINT OUT ALL THE
   *'/' ALPHA-NUMERIC CHARACTERS OF THE STRING WITHIN ''SINGLE'' QUOTELES00600
   *S.'/' THIS FORM IS USED FOR LABELING THE COMPUTER OUTPUT.    '//' LES00610
   *FOR EXAMPLE :   PRINT ''THE ANSWER IS :''                   '//' LES00620
   *PRODUCES THE RESULT :'//5X,' THE ANSWER IS :'// 5X,' 3. PRINT   BY
   * ITSELF IS USED TO SKIP A LINE ON THE COMPUTER OUTPUT .'//)      LES00640
    READ(5,101,END=307)CARD                                          LES00650
307 WRITE(6,107)                                                     LES00660
107 FORMAT('0',' THE COMPUTER OUTPUT SHEET IS DIVIDED INTO 8 ZONES'/' LES00670
   *EACH 15 COLUMNS WIDE . PRINT ZONES CAN BE SKIPPED BY PUTTING A'/' LES00680
   *BLANK IN THE PRINT LIST . FOR EXAMPLE :'/15X,' PRINT A, B, ,X '/'
   *OUTPUTS THE VALUE OF A IN THE FIRST ZONE , B IN THE SECOND ,  '/' LES00700
   *SKIPS THE THIRD ZONE , AND PUTS X IN THE FOURTH ZONE .       '/' LES00710
   *       ALPHA VARIABLES AND ''STRINGS'' MAY EXTEND OVER SEVERAL'/'LES00720
   * ZONES , BUT NUMERIC RESULTS ARE LEFT ADJUSTED IN THE SPECIFIED'/'LES00730
   * ZONE . IF MORE THAN EIGHT ITEMS OCCUR IN THE PRINT LIST , THE'/'
   *ITEMS WILL OVERFLOW AND BE PRINTED ON THE NEXT LINE .'//)        LES00750
    READ(5,101,END=308)CARD                                          LES00760
308 WRITE(6,108)                                                     LES00770
108 FORMAT('0',' USING THE PRINT AND END STATEMENT YOU NOW HAVE THE'/'LES00780
```

```
   * FACILITY TO EXECUTE YOUR FIRST PROGRAMS . FOR EXAMPLE :  '/10X,' LES00790
   *REM  PROGRAM TO COMPUTE THE SQUARE OF A NUMBER             '/10X,' LES00800
   *PRINT''5 SQUARED ='',5**2'/10X,' END                        '//' LES00810
   *PRODUCES THE RESULT :'//5X,' 5 SQUARED =  25                '//' LES00820
   *YOU WILL NOW BE GIVEN TWO PROBLEMS TO SOLVE . YOU WILL ENTER  '/' LES00830
   *THE EXECUTION PHASE OF CAIBASIC WHERE YOU CAN RUN YOUR PROBLEMS'/'LES00840
   * AND THEN YOU WILL RETURN TO FINISH THE LESSON.'//5X,'1) FIND THE LES00850
   *SQUARE ROOT OF  5 SQUARED MINUS 4 TIMES 2 TIMES 2 .'//5X, '2) FINDLES00860
   * THE VALUE OF  3.1416(B CUBED)H/12'/' WHERE B=2.50 , H=3.03 . '//)
309 WRITE(6,109)                                                    LES00880
109 FORMAT('0',' IF YOU WISH TO SKIP THESE PROBLEMS REPLY : YES   '/'
   *AND THE  LESSON WILL CONTINUE.'//)
  5 READ(5,101,END=309)CARD                                         LES00910
    CALL CRUNCH(ILNGTH)                                             LES00920
    IF(CARDP(1).EQ.ASTRSK)GO TO 5                                   LES00930
    IF(CARDP(1).EQ.ALPHA(25))GO TO 10                               LES00940
    CALL TEST1                                                      LES00950
310 WRITE(6,110)                                                    LES00960
110 FORMAT('0','REPLY WITH THE ANSWER TO THE FIRST PROBLEM .'//)    LES00970
  6 READ(5,101,END=310)CARD                                         LES00980
    CALL CRUNCH(ILNGTH)                                             LES00990
    IF(CARDP(1).EQ.ASTRSK)GO TO 6                                   LES01000
    IF(CARDP(1).EQ.DIGIT(4)) GO TO 7                                LES01010
    WRITE(6,111)                                                    LES01020
111 FORMAT('0',' YOUR ANSWER IS WRONG . YOUR PRINT STATEMENT SHOULD'/'LES01030
   * HAVE BEEN      PRINT ((5**2)-(4*2*2))**.5'//)                  LES01040
  7 WRITE(6,112)                                                    LES01050
112 FORMAT('0',' REPLY WITH THE ANSWER TO THE SECOND PROBLEM .'//)   LES01060
  8 READ(5,101,END=7)CARD                                           LES01070
    CALL CRUNCH(ILNGTH)                                             LES01080
    IF(CARDP(1).NE.DIGIT(2).AND.CARDP(2).NE.DIGIT(3)) GO TO 9       LES01090
    IF(CARDP(3).EQ.DECMAL.AND.CARDP(4).EQ.DIGIT(4)) GO TO 10        LES01100
  9 WRITE(6,113)                                                    LES01110
113 FORMAT('0',' YOUR ANSWER IS WRONG . YOUR PRINT STATEMENT SHOULD'/'LES01120
   * HAVE BEEN      PRINT (3.1416*(2.50**3)*3.03)/12              '//) LES01130
 10 WRITE(6,114)                                                    LES01140
114 FORMAT('0',10X,' C. READ                                 '/5X,' LES01150
   *THE READ STATEMENT IS THE METHOD WHICH PROVIDES INPUT TO THE  '/' LES01160
   *PROGRAM . THE FORM OF THE READ STATEMENT IS :               '//5X,' LES01170
   *READ << VARIABLE,....,VARIABLE >>                          '//' LES01180
   *FOR EVERY VARIABLE IN THE READ LIST THERE MUST BE A CORRESPONDING'LES01190
   */' ELEMENT IN A DATA STATEMENT . THE READ AND DATA STATEMENTS ARE'LES01200
   */' USED TOGETHER TO ASSIGN INPUT VALUES TO PROGRAM VARIABLES.'/ 2X
   *,' WHEN THE READ STATEMENT IS EXECUTED , EACH VARIABLE IS ASSIGNEDLES01220
   *'/' SUCCESSIVE NUMBERS FROM A STACK OF NUMERIC DATA OR SUCCESSIVE'LES01230
   */' ''STRINGS'' FROM A STACK OF ALPHA-NUMERIC DATA. AS EACH VARIABLLES01240
   *E'/' IS READ , IT TAKES THE TOP ELEMENT OF THE APPROPRIATE DATA'/'LES01250
   * STACK .'//)                                                   LES01260
```

```
    READ(5,101,END=315)CARD                                                 LES01270
315 WRITE(6,115)                                                            LES01280
115 FORMAT('0',10X,' D. DATA                                    '/5X,'      LES01290
   *THE DATA STATEMENT IS A LIST OF INPUT NUMBERS OR ''STRINGS'' THAT'LES01300
   *'/' WILL BE ASSIGNED TO VARIABLES IN A READ STATEMENT . THE FORM'/'LES01310
   * OF THE DATA STATEMENT IS :                                    '//5X,' LES01320
   *DATA << NUMBER OR ''STRING'',...., NUMBER OR ''STRING'' >>     '//'    LES01330
   *THE ''STRING'' OF ALPHA-NUMERIC CHARACTERS MUST BE ENCLOSED IN'/'      LES01340
   *SINGLE QUOTES .                                               '/ 5X,' LES
   *DATA STATEMENTS MAY BE PLACED ANYWHERE IN A PROGRAM , BUT THERE'/'LES01360
   * IS AN UPPER LIMIT OF 500 NUMERIC AND 500 ALPHA-NUMERIC DATA   '/'    LES01370
   *ELEMENTS FOR EACH PROGRAM .'//)                                        LES01380
    READ(5,101,END=316)CARD                                                LES01390
316 WRITE(6,116)                                                           LES01400
116 FORMAT('0',10X,' WHEN THE FIRST DATA STATEMENT IS INTERPRETED '/'      LES01410
   *BY THE CAIBASIC COMPILER A FIRST IN , FIRST OUT STACK IS FORMED'/'LES01420
   * FOR NUMERIC AND ALPHA-NUMERIC DATA . AS EACH NUMBER OR STRING'/'     LES01430
   *IN A DATA LIST IS INTERPRETED , IT IS PLACED ON THE BOTTOM OF '/'     LES01440
   *ITS RESPECTIVE DATA STACK . AS OTHER DATA STATEMENTS ARE LOCATED'/LES01450
   *' IN THE PROGRAM ITS ELEMENTS ARE PLACED ON THE BOTTOM OF THE '/'     LES01460
   *PROPER STACK .'/10X,' 1.) EXAMPLE :'//5X,' DATA 10.0,13.33,''J.E.SLES01470
   *MITH'',18'/5X,' DATA ''Z.X. DOE'',19                          '//'    LES01480
   *PRODUCES THE FOLLOWING DATA STACKS :'/5X,'NUMERIC',10X,' ALPHA-NUMLES01490
   *ERIC'/5X,' 10.0',10X,' J.E.SMITH'/5X,' 13.33',10X,' Z.X. DOE'/5X,'LES01500
   * 18'/5X,' 19'//)                                                      LES01510
    READ(5,101,END=317)CARD                                                LES01520
317 WRITE(6,117)                                                           LES01530
117 FORMAT('0',10X,' DURING EXECUTION OF READ STATEMENTS , AS THE '/'      LES01540
   *VARIABLES IN THE READ LIST ARE ASSIGNED VALUES FROM THE TOP OF'/'     LES01550
   *THE APPROPIRATE DATA STACK , THE DATA STACK IS DECREMENTED AND'/'     LES01560
   *THE NEXT ELEMENT POPS UP .'/10X,' 2.) EXAMPLE :'//5X,' READ A,B1,ZLES01570
   *$'//' ASSUMING THAT THE DATA FROM EXAMPLE 1.) IS AVAILABLE , THE 'LES01580
   *'/' VARIABLES IN THE READ LIST ARE ASSIGNED VALUES AS FOLLOWS :'/5XLES01590
   *,'  A <-- 10.0'/5X,' B1 <-- 13.33'/5X,' Z$ <-- J.E.SMITH      '//'  LES01600
   *THE RESULTING DATA STACKS ARE AS FOLLOWS :'/5X,' NUMERIC',10X,' ALLES01610
   *PHA-NUMERIC'/5X' 18',15X,' Z.X. DOE'/5X,' 19'//)                      LES01620
    READ(5,101,END=318)CARD                                                LES01630
318 WRITE(6,118)                                                           LES01640
118 FORMAT('0',10X,' E. RESTORE , RESTORE$                      '//'       LES01650
   *THE RESTORE , RESTORE$  STATEMENTS ARE USED TO RETURN THE NUMERIC'LES01660
   *'/' AND ALPHA-NUMERIC DATA STACKS TO THEIR ORIGINAL CONDITION SO'/'LES01670
   * THAT THE DATA MAY BE USED AGAIN . THE FORM OF THE RESTORE STATEMELES01680
   *NT'/' IS :'//10X,' RESTORE    ( RESTORES NUMERIC DATA )'//10X,' RESLES01690
   *TORE$    ( RESTORES ALPHA-NUMERIC DATA )                      '//'    LES01700
   *YOU WILL NOW BE ASKED SOME QUESTIONS ABOUT THE READ , DATA AND'/'      LES01710
   *RESTORE STATEMENTS :'//10X,' CONSIDER THE FOLLOWING STATEMENTS AS LES01720
   *PART OF A BASIC PROGRAM:'//5X,' DATA 2,5,7,16,''ANS='''/5X,' DATA LES01730
   *''CORRECT'',25,''WRONG'',0.0'/5X,' READ A,B3,I$,C1,D'/5X,' READ J$LES01740
```

```
   *,E4,K$,X'//)                                                        LES01750
    READ(5,101,END=319)CARD                                             LES01760
319 WRITE(6,119)                                                        LES01770
119 FORMAT('0',10X,' 1.) WHAT IS THE VALUE OF C1 ?? REPLY WITH VALUE'/LES01780
   */)                                                                  LES01790
 15 READ(5,101,END=320)CARD                                             LES01800
    CALL CRUNCH(ILNGTH)                                                 LES01810
    IF(CARDP(1).EQ.ASTRSK) GO TO 15                                     LES01820
    IF(CARDP(1).EQ.DIGIT(8)) GO TO 20                                   LES01830
320 WRITE(6,120)                                                        LES01840
120 FORMAT('0',10X,' THE CORRECT ANSWER IS C1=7 . THE DATA IS PUT '/'  LES01850
   *INTO TWO FIFO STACKS , ONE FOR NUMERIC DATA , AND ONE FOR ALPHA-'/LES01860
   *' NUMERIC DATA . WHEN THE READ STATEMENTS ARE EXECUTED , THE   '/'  LES01870
   *VARIABLES ARE ASSIGNED AS FOLLOWS :'//5X' A <-- 2',10X,' I$ <-- ANLES01880
   *S='/5X,' B3 <-- 5',10X,' J$ <-- CORRECT'/5X,' C1 <-- 7',10X,' K$ <LES01890
   *-- WRONG'/5X,' D <-- 16'/5X,' E4 <-- 25'/5X,' X <-- 0.0'//)       LES01900
 20 WRITE(6,121)                                                        LES01910
121 FORMAT('0',10X,' NOW CONSIDER THAT THE FOLLOWING STATEMENTS   '/'  LES01920
   *WERE ADDED TO THE ABOVE PROGRAM :'/5X,' RESTORE'/5X,' READ Z4,A'//LES01930
   *10X,' 2.) WHAT IS THE VALUE OF Z4 ?? REPLY WITH VALUE .'//)       LES01940
 25 READ(5,101,END=322)CARD                                             LES01950
    CALL CRUNCH(ILNGTH)                                                 LES01960
    IF(CARDP(1).EQ.ASTRSK) GO TO 25                                     LES01970
    IF(CARDP(1).EQ.DIGIT(3).AND.CARDP(2).EQ.BLANK) GO TO 30
322 WRITE(6,122)                                                        LES01990
122 FORMAT('0',10X,' THE CORRECT ANSWER IS Z4=2 . THE RESTORE COMMAND'LES02000
   */' RETURNS THE NUMERIC DATA STACK TO ITS ORIGINAL CONDITION, AND'/LES02010
   *' THE READ STATEMENT ASSIGNES VALUES FROM THE TOP OF THE DATA '/' LES02020
   *STACK TO THE VARIABLES IN THE READ LIST AS FOLLOWS :'//5X,' Z4 <--LES02030
   * 2'/5X,' A  <-- 5'//)                                                LES0204
 30 WRITE(6,123)                                                        LES02050
123 FORMAT('0',10X,' F. SUMMARY                                  '/'  LES02060
   *NOW THAT YOU HAVE SEEN HOW READ AND DATA STATEMENTS WORK TOGETHER'LES02070
   */' TO INPUT VALUES INTO YOUR PROGRAM , AND HOW THE PRINT STATEMENTLES02080
   *'/' IS USED TO OUTPUT AND LABEL RESULTS YOU HAVE THE FACILITY '/' LES02090
   *TO WRITE SIMPLE PROGRAMS USING INPUT DATA .'/5X,' FOR EXAMPLE '/'
   *  (( 8**2)**.5) COULD BE WRITTEN IN SYMBOLIC FORM AND THE DATA '/'
   *COULD BE READ IN AS FOLLOWS:'//5X,' READ A,B'/5X,' PRINT''ANS='',(LES02130
   *( A**B)**.5)'/5X,' DATA 8,2'/5X,' END'//' IN THE NEXT LESSON YOU WLES02140
   *ILL BE SHOWN HOW TO USE ASSIGNMENT STATEMENTS'/' THIS WILL GIVE
   *YOU GREATER FLEXIBILITY IN WRITING EXPRESSIONS                   '/'
   *AND WILL ALLOW YOU TO DO ASSIGNMENTS SUCH AS :'//5X,'A=B**2-4*A*C LES02170
   *, AND Z=(A**2+3)/A , THEN BY SAYING PRINT A,Z '/' THE RESULTS OF BLES02180
   *OTH EXPRESSIONS WOULD BE DISPLAYED.'//)                              LES02190
    READ(5,101,END=324)CARD                                             LES02200
324 WRITE(6,124)                                                        LES02210
124 FORMAT('0',10X,' YOU WILL NOW BE GIVEN SOME REPRESENTATIVE PROBLEMLES02220
   *S'/' TO GIVE YOU A CHANCE TO EXERCISE YOUR NEW PROGRAMMING TOOLS'/LES02230
```

```
     */10X,'1.) WRITE A PROGRAM TO SOLVE THE EQUATION  X**2+10Y-24  '/' LES02240
     *WHERE THE INPUT DATA  IS  X=10 ,Y=3 .'//10X,' 2.) WRITE A PROGRAM LES02250
     *TO SOLVE THE QUADRATIC EQUATION'//' (-B+(B**2-4AC)**.5)/2A    '/'
     *INPUT DATA IS  A=2 ,B=5 ,C=2 .'//' IF YOU WISH TO WRITE AND EXECUTLES02270
     *E THESE PROGRAMS NOW , REPLY : YES'/' AND YOU WILL ENTER THE CAIBALES02280
     *SIC COMPILER ; OTHERWISE REPLY : NO"/' AND YOU WILL GO ON TO THE NLES02290
     *EXT LESSON .'//)                                                  LES02300
  35 READ(5,101,END=325)CARD                                           LES02310
     CALL CRUNCH(ILNGTH)                                               LES02320
     IF(CARDP(1).EQ.ASTRSK) GO TO 35                                   LES02330
     IF(CARDP(1).EQ.ALPHA(25))GO TO 45                                 LES02340
     IF(CARDP(1).EQ.ALPHA(14))GO TO 40                                 LES02350
 325 WRITE(6,125)                                                      LES02360
 125 FORMAT('0',' *** YOUR REPLY IS INCORRECTLY TYPED , REPLY AGAIN ***LES02370
    *'//)                                                              LES02380
  40 WRITE(6,126)                                                      LES02390
 126 FORMAT('0',' IF YOU DECIDE TO RUN THESE PROBLEMS LATER THE ANSWERSLES02400
    *'/' WILL NOW BE GIVEN SO YOU MAY CHECK YOUR RESULTS :'//5X,' 1.)  LES02410
    *106'/5X,' 2.) -.500'//)                                           LES02420
     CALL EXIT
  45 CALL TEST1                                                        LES02440
     WRITE(6,127)                                                      LES02450
 127 FORMAT('0',10X,' REPLY WITH YOUR ANSWER TO PROBLEM 1.)'//)        LES02460
  50 READ(5,101,END=328)CARD                                           LES02470
     IF(CARDP(1).EQ.ASTRSK)GO TO 50                                    LES02480
     IF(CARDP(1).EQ.DIGIT(2).AND.CARDP(3).EQ.DIGIT(7)) GO TO 55        LES02490
 328 WRITE(6,128)                                                      LES02500
 128 FORMAT('0',' THE CORRECT ANSWER IS 106 AND THE PROGRAM SHOULD '/' LES02510
    *HAVE BEEN SIMILAR TO :'//5X,' READ X,Y'/5X,' PRINT''ANSWER='',(X**LES02520
    *2 + 10*Y - 24)'/5X,' DATA 10,3'/5X,' END'//)                      LES02530
  55 WRITE(6,129)                                                      LES02540
 129 FORMAT('0',10X,' REPLY WITH YOUR ANSWER TO PROGRAM 2.)'//)        LES02550
  60 READ(5,101,END=62)CARD                                            LES02560
     CALL CRUNCH(ILNGTH)                                               LES02570
     IF(CARDP(1).EQ.ASTRSK) GO TO 60                                   LES02580
     IF(CARDP(1).NE.CMINUS.AND.CARDP(2).NE.DECMAL)GO TO 62             LES02590
     IF(CARDP(3).EQ.DIGIT(6))GO TO 70                                  LES02600
  62 WRITE(6,130)                                                      LES02610
 130 FORMAT('0',10X,' THE CORRECT ANSWER IS -.50 AND THE PROGRAM SHOULDLES02620
    *'/' LOOK SIMILAR TO :'//5X,' READ A,B,C'/5X,' PRINT''ANSWER='',(-BLES02630
    * + (B**2 - 4*A*C)**.5) / 2*A'/5X,' DATA 2,5,2'/5X,'  END'//)      LES02640
  70 CALL EXIT
     END                                                               LES02660
C
CC    LESSON 3 PRESENTS THE 'LET' STATEMENT AND BUILT-IN FUNCTIONS
C
                              COMMON                                   LES04010
     -         STACK(100), PROG(2000), CARD(80), CARDP(80), ALPHA(48),  LES04020
```

```
     -          IAPTR, INPTR, IADATA(500) , XNDATA(500) , STRING(5),      LES04030
     -          DIGIT(10), IPRITB(10), LISTST(100), ISTLST(100),          LES04040
     -          PRT(2500), NERRS, INST, NSTLST, DEBUG, DOLSGN, QUOTE,     LES04050
     -          EQUALS, PARRT, DECMAL, PLUS, CMINUS, SLASH, COMMA,        LES04060
     -          PARLFT, ASTRSK, BLANK                                     LES04070
      COMMON  INTERP,IEXERR                                              LES04080
      REAL*8 LES3/'LESON3  '/
      CALL ALOAD(LES3,N1)
      WRITE(6,100)
  100 FORMAT('0',5X,' *** LESSON 3. ***                                  '//'
     *THIS INSTRUCTION SET WILL INTRODUCE YOU TO ASSIGNMENT STATEMENTS'/
     *' AND BUILT-IN FUNCTIONS.THE LET STATEMENT AND THE 10 BUILT IN'/'
     *FUNCTION WILL ENABLE YOU TO EVALUATE AND ASSIGN VARIABLES TO '/'
     *COMPLEX ARITHMETIC EXPRESSIONS.'//)
      READ(5,101,END=302)CARD
  101 FORMAT(80A1)
  302 WRITE(6,102)
  102 FORMAT('0',10X,' A. LET                                            '/5X,'
     *THE LET STATEMENT IS AN ASSIGNMENT OR SUBSTITUTION COMMAND. IT'/'
     *CAUSES THE EVALUATION OF AN EXPRESSION TO BE SUBSTITUTED FOR THE'/
     *' CURRENT VALUE OF A VARIABLE.THE FORM OF THE LET STATEMENT IS:'//
     *10X,' LET << VARIABLE >> = << EXPRESSION >>      OR      '//10X,'
     *LET << VARIABLE >> = << VARIABLE >> = .... = << EXPRESSION >>'//)
      READ(5,101,END=303)CARD
  303 WRITE(6 ,103)
  103 FORMAT('0',5X,' THERE MAY BE ANY NUMBER OF VARIABLE = VARIABLE'/'
     *IN THE FORM OF THE LET STATEMENT . AN EXPRESSION IS A NUMBER ,'/'
     * VARIABLE , OR AN ARITHMETIC EXPRESSION .                 '//10X,'
     *WHEN THE LET STATEMENT IS EXECUTED THE EXPRESSION ON THE RIGHT'/'
     *SIDE OF THE EQUAL SIGN IS EVALUATED AND THE RESULTING VALUE IS'/'
     *ASSIGNED TO THE VARIABLE OR VARIABLES ON THE LEFT SIDE OF THE EQUA
     *L'/' SIGN . THE PREVIOUS VALUE ASSIGNED TO THE VARIABLE OR VARIABL
     *ES'/' WILL BE LOST . FOR EXAMPLE :'//10X,' LET A=12.3'/10X,' LET B
     *=14.4'/10X,' LET A=B=25'//' THE VALUE OF THE VARIABLES A AND B IS
     *NOW A=B= 25 .'//)
      READ(5,101,END=304)CARD
  304 WRITE(6,104)
  104 FORMAT('0',10X,' THE ONLY RESTRICTION ON THE USE OF VARIABLES '/'
     *IS THAT  SIMPLE AND SUBSCRIPTED VARIABLES CAN ONLY BE ASSIGNED'/'
     *NUMERIC VALUES , AND ALPHA VARIABLES CAN ONLY BE ASSIGNED ALPHA-'/
     *' NUMERIC''STRINGS''. FOR EXAMPLE :'//10X,' LET A=D4=X(5)=(3**2 -
     *6)/4'/10X,' LET A$=X$=''HELP'''/10X,' LET Y=(B - 4*A*C)**.5'/10X'
     *LET O$=''ANSWER ='''/' THE FOLLOWING ASSIGNMENT IS ILLEGAL :'//10X
     *' LET A=D$=(3+4)'//)
      READ(5,101,END=305)CARD
  305  WRITE(6,105)
  105 FORMAT('0',10X,' YOU WILL NOW BE ASKED QUESTIONS CONCERNING   '/'
     *WHAT YOU HAVE JUST LEARNED :'//)
```

```
    WRITE(6,106)
106 FORMAT('0',10X,' 1.)  REPLY WITH VALUE OF X IN BELOW PROGRAM .'//1
   *0X,' LET A=4'/10X,' LET B=6'/10X,' LET C=3'/10X,' LET X=A**2 - 4*B
   * + 3*C'/10X,' PRINT''ANSWER ='',X'/10X,' END'//)
  5 READ(5,101,END=306) CARD
    CALL CRUNCH(ILNGTH)
    IF(CARDP(1).EQ.ASTRSK) GO TO 5
    IF(CARDP(1).EQ.DIGIT(2)) GO TO 10
    IF(CARDP(1).EQ.PLUS.AND.CARDP(2).EQ.DIGIT(2)) GO TO 10
306 WRITE(6,107)
107 FORMAT('0',' YOUR RESPONSE WAS INCORRECT . THE EXPRESSION TO  '/'
   *EVALUATED IS :  X=4**2 - 4*6 + 3*3  WHICH EVALUATES AS :'15X,' X <
   *-- 16 - 24 + 9   ;  X <-- + 1'//)
 10 WRITE(6,108)
108 FORMAT('0',10X, ' 2.) REPLY WITH VALUE OF Z IN BELOW PROGRAM .'//1
   *0X,' DATA 1,2,3,4,6'/10X,' DATA''RIGHT ON'',7,8'/10X,' READ A,B,X'
   */10X,' READ G$,Y,C,D '/10X,' LET Z=X+Y'/10X,' END'//)
 15 READ(5,101,END=309)CARD
    CALL CRUNCH(ILNGTH)
    IF(CARDP(1).EQ.ASTRSK) GO TO 15
    IF(CARDP(1).EQ.DIGIT(8)) GO TO 20
309 WRITE(6,109)
109 FORMAT('0',' YOUR RESPONSE WAS INCORRECT . THE VARIABLES ARE  '/'
   *ASSIGNED VALUES AS FOLLOWS :'/5X,' NUMERIC',10X,' ALPHA'/ 5X,'  A<
   *-- 1',10X,' Y$<-- RIGHT ON'/5X,' B<-- 2'/5X,' X<-- 3'/5X,' Y<-- 4'
   */5X,' C<-- 6'/5X,' D<--7'//5X,' Z <-- X+Y   ;  Z <-- 7'//)
 20 WRITE(6,110)
110 FORMAT('0',10X,' 3.)  LET R= A+B/C-D'/' WHICH FORMULA DOES THIS
   * STATEMENT REPRESENT ?'//10X,' REPLY WITH CORRECT LETTER'/10X,' A.
   *) R=(A+B)/(C-D)'/10X,' B.) R= A+(B/C)-D'//)
 25 READ(5,101,END=311)CARD
    CALL CRUNCH(ILNGTH)
    IF(CARDP(1).EQ.ASTRSK) GO TO 25
    IF(CARDP(1).EQ.ALPHA(2)) GO TO 30
311 WRITE(6,111)
111 FORMAT('0',' YOUR RESPONSE WAS INCORRECT . WHEN THERE ARE NO  '/'
   *PARENTHESIS IN AN EXPRESSION , THE HIERARCHY OF OPERATORS APPLIES'
   */' THUS IN THIS EXPRESSION THE DIVIDE OPERATION IS DONE FIRST,'/'
   *THEN ADDITION AND SUBTRACTION . IF YOU ARE STILL HAVING PROBLEMS'/
   *' WITH EXPRESSIONS YOU HAD BETTER REVIEW YOUR SESSION ON LESSON 1.
   *'//)
 30 WRITE(6,112)
112 FORMAT('0',10X,' B. BUILT-IN FUNCTIONS                          '/5X,'
   *BUILT-IN FUNCTIONS ARE COMMONLY USED PROGRAMS ALREADY WRITTEN '/'
   *AND STORED IN THE CAIBASIC COMPILER FOR YOUR USE . THERE ARE  '/'
   *FUNCTIONS TO FIND SQUARE ROOTS , LOGARITHMS , ABSOLUTE VALUES ,'/'
   * AND TRIGONOMETRIC VALUES . THE FORM FOR THE BUILT-IN FUNCTIONS IS
   * :'//10X,' FUNCTION NAME << (EXPRESSION) >>'/' WHERE THE EXPRESSIO
```

```
   *N IS ENCLOSED IN PARENTHESIS .'//)
    READ(5,101,END=313)CARD
313 WRITE(6,113)
113 FORMAT('0',10X,' THE BUILT-IN FUNCTIONS AND DEFINITIONS ARE :'//5X
   *,' SQR(X)     --- SQUARE ROOT OF ARGUMENT (MUST BE POSITIVE)'/5X,'
   * ABS(X)      --- ABSOLUTE VALUE OF ARGUMENT'/5X,' LOG(X)      ---
   *NATURAL LOGARITHM OF ARGUMENT '/5X,' EXP(X)      --- EXPONENTIAL FU
   *NCTION , VALUE OF 2.718218 ** X'/5X,' INT(X)      --- INTEGER PART
   * OF ARGUMENT IS RETURNED                          '/5X,' SIN(X)      -
   *-- SINE OF THE ARGUMENT'/5X,' COS(X)      --- COSINE OF ARGUMENT'/
   *5X,' TAN(X)      --- TANGENT OF ARGUMENT'/5X,' ATN(X)      --- ARC
   *TANGENT IN RADIANS OF ARGUMENT'//' IN ALL THE ABOVE BUILT-IN FUNCT
   *IONS THE ARGUMENT IS ANY LEGAL EXPRESSION ;'/' AND AS NOTED THE SQ
   *R FUNCTION  REQUIRES A POSITIVE ARGUMENT ,'/' THE TRIGONOMETRIC FU
   *NCTIONS REQUIRE AN ARGUMENT VALUE IN RADIANS .'//)
    READ(5,101,END=314)CARD
314 WRITE(6,114)
114 FORMAT('0',10X,' THE BUILT-IN FUNCTIONS ARE USED BY SIMPLY    '/'
   *CALLING THEM WITH THE APPROPRIATE FUNCTION NAME AND ARGUMENT .'/'
   *THESE BUILT-IN FUNCTIONS ARE CONSIDERED TO BE EXPRESSIONS ,    '/'
   *AND THEY MAY BE USED ANY PLACE WHERE AN EXPRESSION IS LEGAL ,'/'
   *FOR EXAMPLE :'//5X,' LET Z=SQR(ABS(-5))    IS A CORRECT USE OF BUI
   *LT IN FUNCTIONS.'//)
    READ(5,101,END=315)CARD
315 WRITE(6,115)
115 FORMAT('0',10X,' THE FOLLOWING PROGRAM IS A EXAMPLE OF HOW TO '/'
   *USE BUILT-IN FUNCTIONS :'//5X,' REM  PROGRAM TO COMPUTE SQUARE ROO
   *T AND LOGARITHMS'/5X,' READ A'/5X,' LET Y=SQR(A)'/5X,' LET Z=LOG(A
   *)'/5X,' PRINT''SQUARE ROOT='',Y,''LOG='',Z'/5X,' DATA 4,6,8'/5X,'
   *END'//' PRODUCES RESULT'/5X,' SQUARE ROOT= 2.0',10X,' LOG= 1.386'/
   */)
    READ(5,101,END=316)CARD
316 WRITE(6,105)
    WRITE(6,116)
116 FORMAT('0',10X,' 1.) IS THE FOLLOWING STATEMENT LEGAL'/' REPLY : Y
   *ES , OR  NO.'//5X,' LET Z1=4*SQR(3.0 * EXP(9.3))'//)
 35 READ(5,101,END=317)CARD
    CALL CRUNCH(ILNGTH)
    IF(CARDP(1).EQ.ASTRSK) GO TO 35
    IF(CARDP(1).EQ.ALPHA(25)) GO TO 40
317 WRITE(6,117)
117 FORMAT('0',' YOUR REPLY WAS INCORRECT . MULTIPLE FUNCTIONS CAN'/'
   *BE USED IN AN EXPRESSION AS LONG AS THE EXPRESSION IS WELL FORMED'
   *//)
 40 WRITE(6,118)
118 FORMAT('0',10X,' 2. IS THE FOLLOWING SEQUENCE OF PROGRAM STATEMENT
   $S LEGAL'/' REPLY: YES OR NO.'//5X,' LET B=-9'/5X,' LET X=SQR(B)'/1
   *0X,' ...'/10X,' ...'/10X,' ...'//)
```

```
 50 READ(5,101,END=319)CARD
    IF(CARDP(1).EQ.ASTRSK) GO TO 50
319 WRITE(6,119)
119 FORMAT('0',' NO , THE SQUARE ROOT OF A NEGATIVE NUMBER IS AN  '/'
   *UNDEFINED OPERATION . IN THE NEXT LESSON YOU WILL BE SHOW A   '/'
   *BASIC STATEMENT FOR TESTING AND BRANCHING TO ANOTHER SEGMENT  '/'
   *OF THE PROGRAM IF THE TEST IS TRUE . FOR EXAMPLE THE ABOVE    '/'
   *PROGRAM SEQUENCE MIGHT BE ALTERED AS FOLLOWS :'//5X,' LET B=-9'/5X
   *,' IF B LT 0 THEN 100'/5X,' 50 LET X=SQR(X)'/10X,' ...'/10X,' ...'
   */10X,' ...'/5X,' 100 REM  NEGATIVE ARGUMENT'/5X,' LET B=ABS(B)'/5X
   *,' GO TO 50'/10X,' ...'/10X,' ...'//' THIS PROGRAM SEQUENCE TESTS
   *FOR A NEGATIVE ARGUMENT , IF TRUE'/' IT BRANCHES TO STATEMENT NUMB
   *ER 100 , MAKES THE ARGUMENT POSITIVE'/' AND BRANCHES BACK TO STATE
   *MENT 50  TO COMPLETE THE PROGRAM .'//)
    READ(5,101,END=320)CARD
320 WRITE(6,123)
123 FORMAT('0',10X,' C. SUMMARY                                      '/'
   *WITH THE LET STATEMENT AND BUILT-IN FUNCTIONS , PLUS THE PREVIOUS'
   $/' BASIC STATEMENTS ( REM , READ , DATA , PRINT ) , YOU ARE   '/'
   $FAST GAINING AN EFFECTIVE REPERTOIRE FOR PROGRAMMING USE . IN '/'
   *THE NEXT LESSON YOU WILL LEARN HOW TO SET UP LOOPS IN A PROGRAM'/'
   *  SO THAT THE MAIN BODY OF A PROGRAM MAY BE EXECUTED AS OFTEN  '/'
   *AS DESIRED . AS YOU ARE DOING YOUR REVIEW PROBLEMS , THINK ABOUT'/
   *' HOW YOU COULD SET UP A LOOP TO READ IN ANY AMOUNT OF DATA , '/'
   *PROCESS IT AND THEN HALT FOR SOME TEST CONDITION .'//)
    READ(5,101,END=324) CARD
324 WRITE(6,124)
124 FORMAT('0',10X,' THE FOLLOWING REVIEW PROBLEMS WILL EXERCISE  '/'
   *YOUR PROGRAMMING SKILLS TO DATE :'//1X,' 1.) WRITE A PROGRAM TO CO
   *MPUTE THE PRESENT WORTH OF AN INVESTMENT FOR'/' SOME NUMBER OF YEA
   *RS HENCE . THE FORMULA IS :'/10X,' P=S( 1/((1+I)**N))'/' WHERE  P
   $ IS THE PRESENT WORTH OF A PAYMENT  S  IN  N  YEARS HENCE'/' AT AN
   *  INTEREST RATE OF  I  . FOR DATA USE I=.08 , S=5000 , N=20 .'//5X,
   *' 2.) WRITE A PROGRAM TO FIND SIDES A , AND C OF A TRIANGLE USING'
   */' THE LAW OF SINES FORMULA :'/10X,' A/SIN(A) = B/SIN(B) = C/SIN(C
   *)'/' WHERE THE NUMERATOR IS THE SIDE AND THE DENOMIATOR IS THE'/'
   *SINE OF THE ANGLE , THE CONVERSION FACTOR FROM DEGREES TO RADIANS
   *IS:'/10X,' 1 DEGREE =(3.1416/180) RADIANS . THE DATA FOR THE PROGR
   *AM IS :'/5X,' SIDE B=34.91 IN.',5X,' ANGLE A=98.71 DEG.'/5X,' ANGL
   *E B=49.97 DEG.',5X,' ANGLE C=31.32 DEG.'//' IF YOU WISH TO RUN THE
    CALL CRUNCH(ILNGTH)
    IF(CARDP(1).EQ.ASTRSK) GO TO 60
    IF(CARDP(1).EQ. ALPHA(14)) GO TO 65
    IF(CARDP(1).EQ. ALPHA(25)) GO TO 70
325 WRITE(6,125)
125 FORMAT('0',' YOUR REPLY IS INCORRECT , REPLY AGAIN .'//)
    GO TO 60
 65 WRITE(6,126)
```

```
  126 FORMAT('0',' THE ANSWERS TO THE PROBLEMS WILL NOW BE GIVEN SO '/'
     *THAT YOU MAY CHECK YOUR RESULTS LATER :'//5X,'1.) $1072.74'/5X,' 2
     *.) SIDE A=45.06 IN. AND SIDE C=23.69 IN.'//)
      CALL EXIT
   70 CALL TEST1
      WRITE(6,127)
  127 FORMAT('0',10X,' REPLY WITH ANSWER TO QUESTION 1.) ??'//) .
   75 READ(5,101,END=80) CARD
      CALL CRUNCH(ILNGTH)
      IF(CARDP(1).EQ.ASTRSK) GO TO 75
      IF(CARDP(1).NE.DIGIT(2) .OR.CARDP(4).NE.DIGIT(3)) GO TO 80
      IF(CARDP(5).EQ.DECMAL.AND.CARDP(6).EQ.DIGIT(8)) GO TO 90
   80 WRITE(6,128)
  128 FORMAT('0',10X,' THE CORRECT ANSWER IS $1075.74 AND THE PROGRAM'/'
     * SHOULD HAVE BEEN SIMILAR TO :'//5X,' READ S,I,N'/5X,' LET P=S*(1/
     *((1+I)**N))'/5X,' PRINT''PRESENT WORTH='',P'/5X' DATA 5000,.08,20'
     */5X,' END'//)
   90 WRITE(6,129)
  129 FORMAT('0',10X,' REPLY WITH ANSWER TO SIDE A FOR QUESTION 2.)'//)
   91 READ(5,101,END=95)CARD
      IF(CARDP(1).EQ.ASTRSK) GO TO 91
      IF(CARDP(1).NE.DIGIT(5).OR.CARDP(2).NE.DIGIT(6)) GO TO 95
      IF(CARDP(3).EQ.DECMAL.AND.CARDP(4).EQ.DIGIT(1)) GO TO 97
   95 WRITE(6,130)
  130 FORMAT('0',10X,' THE CORRECT ANSWER IS SIDE A=45.06 IN. AND   '/'
     *SIDE C=23.6. IN. . THE PROGRAM SHOULD HAVE BEEN SIMILAR TO:'//5X,
     *' READ A,B,C,B1'/5X,' LET A1=(B1*SIN(A*(3.1416/180)))/SIN(B*3.141
     *6/180))'/5X,' LET C1=(B1*SIN(C*(3.1416/180)))/SIN(B*(3.1416/180))'
     */5X,' PRINT''SIDE A =='',A1,''SIDE C =='',C1'/5X,' DATA 98.71,49.97,
     *31.32,34.91'/5X,' END'//)
   97 CALL EXIT
      END
C
CC    THIS LESSON INTRCDUCES BRANCHING , BOTH CONDITIONAL AND
CCC   UNCONDITIONAL .
C
                              COMMON                                    LES00030
     -          STACK(100), PROG(2000), CARD(80), CARDP(80), ALPHA(48),  LES00040
     -          IAPTR, INPTR, IADATA(500) , XNDATA(500) , STRING(5),     LES00050
     -          DIGIT(10), IPRITB(10), LISTST(100), ISTLST(100),         LES00060
     -          PRT(2500), NERRS, INST, NSTLST, DEBUG, DOLSGN, QUOTE,    LES00070
     -          EQUALS, PARRT, DECMAL, PLUS, CMINUS, SLASH, COMMA,       LES00080
     -          PARLFT, ASTRSK, BLANK                                    LES00090
      COMMON  INTERP,IEXERR                                             LES00100
      REAL*8 LES4/'LESON4  '/
      CALL ALOAD(LES4,N1)
      WRITE(6,100)                                                      LES00110
  100 FORMAT('0',5X,'  *** LESSON 4 ***                          '//' LES00120
```

```
   *THIS INSTRUCTION SEQUENCE WILL COVER BRANCHES . AS YOU HAVE   '/' LES00130
   *SEEN FROM PREVIOUS PROGRAMS , A PROGRAM''S EXECUTION USUALLY  '/' LES00140
   *TAKES A DIRECT ROUTE FROM THE FIRST TO THE LAST STATEMENT .   '/' LES00150
   *THE CONDITION THAT ALLOWS DETOURS TO OCCUR IN PROGRAMS IS CALLED'/LES00160
   *' BRANCHING . THERE ARE TWO TYPES OF BRANCHING : UNCONDITIONAL'/' LES00170
   *AND CONDITIONAL .'//)                                           LES00180
    READ(5,101,END=302)CARD                                        LES00190
101 FORMAT(80A1)                                                   LES00200
302 WRITE(6,102)                                                   LES00210
102 FORMAT('0',5X,' A. UNCONDITIONAL BRANCHES                '/5X,' LES00220
   *AN UNCONDITIONAL BRANCH IS AN IMPERATIVE TRANSFER OF CONTROL  '/' LES00230
   *FROM ONE POINT IN A PROGRAM TO ANOTHER . THERE ARE TWO FORMS  '/' LES00240
   *OF THE UNCONDITIONAL BRANCH : THE ''GO TO'' AND THE ''COMPUTED'/' LES00250
   *GO TO''//10X,' 1.) GO TO << STATEMENT NUMBER >> '//' THIS COMMANDLES00260
   * TRANSFERS PROGRAM CONTROL DIRECTLY TO THE STATEMENT NUMBER'/' ANDLES00270
   * CONTINUES EXECUTION FROM THAT POINT . THE ''GO TO''IS USED'/' FORLES00280
   * FORMING LOOPS IN A PROGRAM .'//)                              LES00290
    READ(5,101,END=303)CARD                                        LES00300
303 WRITE(6,103)                                                   LES00310
103 FORMAT('0',' A SAMPLE LOOP FOLLOWS :     '//5X,' REM PROGRAM TO COLES00320
   *MPUTE PRESENT WORTH'/5X,' REM P=INVESTMENT,S=PRINCIPAL,I=INTEREST LES00330
   *RATE,N=NR.YEARS'/5X,' PRINT''INVESTMENT'',''PRINCIPAL'',''INTERESTLES00340
   *'',''NR.YEARS'''/5X,' 10 READ S,I,N'/5X,' LET P=S*(1/((1+I)**N))'/LES00350
   *6X,'PRINTP,S,I,N'/5X,' GO TO 10'/5X,' DATA 5000,.08,20,5000,.08,10LES00360
   *,5000,.06,10'/5X,' END'/' PRODUCES THE OUTPUT :'//' INVESTMENT',10LES00370
   *X,' PRINCIPAL',10X,' INTEREST',10X,' NR.YEARS'/2X,' 1072.74',16X,'LES00380
   *5000',16X,' .08',16X,' 20'/2X,' 2315.97',15X,' 5000',15X,' .08',15LES00390
   *X,' 10'/2X,' 2791.99',15X,' 5000',15X,' .06',15X,' 10'//' *** ERROLES00400
   *R , YOU TRIED TO READ MORE NUMERIC DATA THAN YOU PUT IN ***'//)  LES00410
    READ(5,101,END=304)CARD                                        LES00420
304 WRITE(6,104)                                                   LES00430
104 FORMAT('0',' THE ERROR OCCURS BECAUSE YOU RUN OUT OF DATA DURING'/LES00440
   *' THE EXECUTION OF THE LOOP SET-UP BY THE UNCONDITIONAL TRANSFER .LES00450
   *'/' IF THE READ STATEMENT WERE NOT IN THE LOOP TO CAUSE THE   '/' LES00460
   *PROGRAM TO STOP . THEN YOU WOULD BE IN AN ''INFINITE LOOP'' , '/' LES00470
   *A CONDITION IN WHICH THERE IS NO WAY TO STOP , YOU MUST ALWAYS'/' LES00480
   *CHECK FOR THE ''INFINITE LOOP''CONDITION BY MAKING SURE THAT  '/' LES00490
   *YOUR PROGRAM HAS AN EXIT .'//)                                  LES00500
    READ(5,101,END=305)CARD                                        LES00510
305 WRITE(6,105)                                                   LES00520
105 FORMAT('0', 1X,' 2.) ON << EXPRESSION >> GO TO << STATEMENT NUMBER
   * ,,,,,, STATEMENT NUMBER >>'//' THIS SPECIAL FORM OF THE ''GO TO''LES00540
   * COMMAND ,'/' IS CALLED THE ''COMPUTED GO TO'', THE EXPRESSION'/' LES00550
   *IN THE FORM OF THE STATEMENT MUST EVALUATE TO AN INTEGER NUMBER'/'LES00560
   * BETWEEN 1-->9999 . IF IT IS NOT AN AN INTEGER , OR OUTSIDE THIS'/LES00570
   *' RANGE AN ERROR WILL OCCUR .'/5X,' WHEN THE''COMPUTED GO TO''IS ELES00580
   *XECUTED THE EXPRESSION IS EVALUATED ,'/' AND PROGRAM CONTROL TRANSLES00590
   *FERS TO THE N-TH STATEMENT NUMBER ,'/' WHERE N-TH REPRESENTS THE  LES00600
```

```
   *VALUE OF THE EXPRESSION . FOR EXAMPLE :'//5X,' LET I=3'/5X,' ON I LES00610
   *GO TO 100,33,475,9999'//)                                        LES00620
    READ(5,101,END=306)CARD                                          LES00630
306 WRITE(6,106)                                                     LES00640
106 FORMAT('0',' EXECUTION OF THE ''COMPUTED GO TO'' WOULD CAUSE  '/' LES00650
   *PROGRAM CONTROL TO TRANSFER UNCONDITIONALLY TO STATEMENT NUMBER 47LES00660
   *5 .'/5X,' YOU MUST BE CAREFUL WHEN USING THE ''COMPUTED GO TO'''/'LES00670
   * NOT ONLY BECAUSE OF INFINITE LOOPS ; BUT BECAUSE THE EXPRESSION'/LES00680
   *' MUST BE AN INTEGER BETWEEN 1-->9999 , AND THERE MUST BE A   '/' LES00690
   *STATEMENT NUMBER FOR ''ALL'' POSSIBLE VALUES OF THE EXPRESSION .'/LES00700
   */)                                                               LES00710
    READ(5,101,END=307)CARD                                          LES00720
307 WRITE(6,107)                                                     LES00730
107 FORMAT('0',5X,' B. CONDITIONAL BRANCHING                  '/5X,' LES00740
   *THE CONDITIONAL BRANCH TRANSFERS CONTROL ONLY IF CERTAIN     '/' LES00750
   *RELATIONS ARE TRUE . IF THE THE TEST OF RELATIONS IS TRUE THEN'/' LES00760
   *TRANSFER OF CONTROL OCCURS ; OTHERWISE PROGRAM CONTROL CONTINUES'/LES00770
   *' WITH THE NEXT STATEMENT . THE FORM OF THE CONDITIONAL BRANCH IS:LES00780
   *'// 1X,' IF << EXPRESSION >> << RELATION >> << EXPRESSION >> THEN
   *<< STATEMENT NUMBER >>'//' NOTE THAT ALPHA VARIABLES ARE NOT ALLOWLES00800
   *ED AS AN EXPRESSION IN A'/'  CONDITIONAL BRANCH .'//)
    READ(5,101,END=30,)CARD                                          LES00820
308 WRITE(6,108)                                                     LES00830
108 FORMAT('0',5X,' THE RELATIONS ARE :                       '//'  LES00840
   *SYMBOLS',2X,' EXAMPLE',5X,'MEANING                               LES00850
   *'//2X,' GT',5X,' A GT B',5X,' A GREATER THAN B'/2X,' GE',5X,' A GELES00860
   * B',5X,' A GREATER THAN OR EQUAL TO B'/2X,' LT',5X,' A LT B',5X,' LES00870
   *A LESS THAN B'/2X,' LE',5X,' A LE B',5X,' A LESS THAN OR EQUAL TO LES00880
   *B'/2X,' NE',5X,' A NE B',5X,' A NOT EQUAL TO B'/3X,' =',5X,' A = BLES00890
   *',6X,' A EQUAL B'//)                                             LES00900
    READ(5,101,END=309)CARD                                          LES00910
309 WRITE(6,109)                                                     LES00920
109 FORMAT('0',5X,' WHEN THE CONDITIONAL STATEMENT IS EXECUTED THE'/' LES00930
   *( EXPRESSION RELATION EXPRESSION ) IS TESTED , AND IF THE RELATIONLES00940
   *'/' IS TRUE , THEN CONTROL IS TRANSFERRED TO THE STATEMENT NUMBER.LES00950
   *'/' OTHERWISE PROGRAM CONTROL CONTINUES TO THE NEXT SEQUENTIAL'/' LES00960
   *STATEMENT , FOR EXAMPLE :'//5X,' 1 READ A'/5X,' ...'/5X,' ...'/5X,LES00970
   *' IF A LT 0 THEN 90'/5X,' LET X=SQR(A)'/5X,' ...'/5X,' ...'/5X,'90LES00980
   * PRINT''ILLEGAL ARGUMENT'',A'/5X,' GO TO 1'//5X,'...'//' THE ONLY TLES00990
   *IME THAT THE CONDITIONAL BRANCH IS EXECUTED IS WHEN'/' A IS LESS TLES01000
   *HAN 0 .'//)                                                       LES01010
    READ(5,101,END=310)CARD                                          LES01020
310 WRITE(6,110)                                                     LES01030
110 FORMAT('0',' YOU WILL NOW BE ASKED SOME QUESTIONS ABOUT BRANCHING LES01040
   *:'/10X,'1.) ARE THE FOLLOWING STATEMENTS CORRECT'/' REPLY: YES OR
   *NO)'//15X,' GO TO END'//)
  5 READ(5,101,END=311)CARD                                          LES01070
    CALL CRUNCH(ILNGTH)                                              LES01080
```

```
    IF(CARDP(1).EQ.ASTRSK) GO TO 5                                      LES01090
    IF(CARDP(1).EQ.ALPHA(14))GO TO 10                                   LES01100
311 WRITE(6,111)                                                        LES01110
111 FORMAT('0',' YOUR ANSWER IS INCORRECT . THE ONLY THING THAT IS'/'   LES01120
   *ALLOWED AFTER A ''GO TO'' IS A STATEMENT NUMBER BETWEEN 1-->9999'/LES01130
   */)                                                                  LES01140
 10 WRITE(6,112)                                                        LES01150
112 FORMAT('0',15X,' IF X1 LT A$ THEN 10'//)                            LES01160
 15 READ(5,101,END=313)CARD                                             LES01170
    CALL CRUNCH(ILNGTH)                                                 LES01180
    IF(CARDP(1).EQ.ASTRSK) GO TO 15                                     LES01190
    IF(CARDP(1).EQ.ALPHA(14)) GO TO 20                                  LES01200
313 WRITE(6,113)                                                        LES01210
113 FORMAT('0',' YOUR ANSWER IS INCORRECT . ALPHA VARIABLE MAY NOT'/'   LES01220
   *BE USE FOR TESTING PURPOSES IN A CONDITIONAL STATEMENT .'//)        LES01230
 20 WRITE(6,114)                                                        LES01240
114 FORMAT('0',15X,' IF(X**3-4) GE 10 THEN GO TO 100'//)                LES01250
 25 READ(5,101,END=315)CARD                                             LES01260
    CALL CRUNCH(ILNGTH)                                                 LES01270
    IF(CARDP(1).EQ.ASTRSK) GO TO 25                                     LES01280
    IF(CARDP(1).EQ.ALPHA(14)) GO TO 30                                  LES01290
315 WRITE(6,115)                                                        LES01300
115 FORMAT('0',' YOUR ANSWER IS INCORRECT . THE ONLY THING ALLOWED'/'   LES01310
   *AFTER THEN IS A STATEMENT NUMBER .'//)                              LES01320
 30 WRITE(6,116)                                                        LES01330
116 FORMAT('0',15X, 'LET Y=6,'/15X,' ON Y GO TO 1,3,5,7,999'//)         LES01340
 35 READ(5,101,END=317)CARD                                             LES01350
    CALL CRUNCH(ILNGTH)                                                 LES01360
    IF(CARDP(1).EQ.ASTRSK) GO TO 35                                     LES01370
    IF(CARDP(1).EQ.ALPHA(14)) GO TO 40                                  LES01380
317 WRITE(6,117)                                                        LES01390
117 FORMAT('0',' YOUR ANSWER IS INCORRECT . THERE ARE NOT ENOUGH  '/'   LES01400
   *STATEMENT NUMBERS IN THE '' COMPUTED GO TO''.'//)                   LES01410
 40 WRITE(6,118)                                                        LES01420
118 FORMAT('0',10X,' 2.) CONSIDER THIS PROGRAM SEGMENT , ANY ERRORS (RLES01430
   *EPLY: YES OR NO).'//5X,' 1 READ A,B'/5X,' 3 LET Z=A*B'/5X,' IF Z LLES01440
   *T 0 GO TO  999'/5X,' ...'/5X,' ...'/5X,' GO TO 3'/5X,' DATA 3,5,7,LES01450
   *10'/5X,' 999 END'//)                                              LES01460
    READ(5,101,END=319)CARD                                             LES01470
319 WRITE(6,119)                                                        LES01480
119 FORMAT('0',' THERE IS AN ''INFINITE'' LOOP IN THIS PROGRAM . IT'/'LES01490
   * COULD BE CORRECTED BY CHANGING THE GO TO 3 , TO GO TO 1 . '/5X,' LES01500
   * WHEN YOU WRITE PROGRAMS HAVING BRANCHES CR LOOPS YOU MUST ALWAYS'LES01510
   */' CONSIDER HOW THE PROGRAM WILL STOP. YOU HAVE OBSERVED THAT '/' LES01520
   *AN ''INFINITE LOOP''CAN BE STOPPED BY HAVING A READ STATEMENT '/' LES01530
   *IN THE LOOP AND JUST RUN OUT OF DATA . HOWEVER, ENDING A PROGRAM' LES01540
   */' ON AN ERROR IS AN INELIGANT METHOD . THE MOST COMMON METHODS'/'LES01550
   * USE THE ''GO TO'' AND '' IF/THEN'' COMMANDS TO CONTROL PROGRAM'/'LES01560
```

```
   * LOOPS AND ARE AS FOLLOWS :'//)                                          LES01570
    READ(5,101,END=320)CARD                                                 LES01580
320 WRITE(6,120)                                                            LES01590
120 FORMAT('0',5X,' 1. COUNT AND TEST METHOD , IN WHICH A COUNTER '/' LES01600
   *IS INCREMENTED IN THE LOOP , AND WHEN THE COUNTER REACHES A   '/' LES01610
   *CERTAIN VALUE , BRANCH OUT OF THE LOOP . EXAMPLE PROGRAM '//10X,' LES01620
   *REM COUNT AND TEST METHOD'/10X,' REM N IS COUNTER , INITALIZED TO LES01630
   *0 , AND COUNTS FROM 1-->10'/10X,' LET N=Z=0'/10X,' READ X'/10X,' 1LES01640
   *0 LET Z=Z+X '/10X,' LET N=N+1'/10X,' IF N GT 10 THEN 100'/10X,' GOLES01650
   * TO 10'/10X,' 100 PRINT''SUM='',Z'/10X,' DATA 10'/10X,' END'//' RELES01660
   *PLY WITH VALUE OF Z .'//)                                              LES01670
 45 READ(5,101,END=50) CARD                                                LES01680
    CALL CRUNCH(ILNGTH)                                                     LES01690
    IF(CARDP(1).EQ.ASTRSK) GO TO 45                                         LES01700
    IF(CARDP(1).NE.DIGIT(2).OR.CARDP(2).NE.DIGIT(2)) GO TO 50               LES01710
    IF(CARDP(3).EQ.DIGIT(1)) GO TO 60                                       LES01720
 50 WRITE(6,121)                                                            LES01730
121 FORMAT('0',' YOUR ANSWER IS INCORRECT . THE ONLY WAY TO BE    '/' LES01740
   *SURE OF VALUES IN A LOOP IS TO SET UP A TABLE OF THE VARIABLES'/' LES01750
   *AND KEEP TRACK OF THERE VALUES IN THE LOOP :'/5X,'X',5X,'N',5X,'Z'LES01760
   */4X,'10',4X,'0',5X,'0'/10X,'1',5X,'10'/10X,'2',5X,'20'/10X,'3',5X,LES01770
   *'30'/10X' ...'/10X,' ...'/10X,'9',5X,'90'/10X,'10',4X,'100'/10X,'1LES01780
   *1',4X,'110'//' THUS Z=110'//)                                          LES01790
 60 WRITE(6,122)                                                            LES01800
122 FORMAT('0',5X,' 2. READ AND TEST METHOD , IN WHICH A VALUE IS '/' LES01810
   *READ IN AND TESTED FOR THE END OF LOOP CONDITION .       '//10X,' LES01820
   *REM READ AND TEST DEMONSTRATION'/10X,' REM 9999 IS END VALUE'/10X,LES01830
   *' LET Z=0'/10X,' 1 READ X'/10X,' IF X=9999 THEN 9999'/10X,' LET Z=LES01840
   *Z+X'/10X,' GO TO 1'/10X,' DATA 1,2,3,4,5,6,9999'/10X,' 9999 PRINT LES01850
   *''SUM='',Z'/10X,' END'//' REPLY WITH VALUE OF Z'//)                    LES01860
 65 READ(5,101,END=323)CARD                                                LES01870
    CALL CRUNCH(ILNGTH)                                                     LES01880
    IF(CARDP(1).EQ.ASTRSK) GO TO 65                                         LES01890
    IF(CARDP(1).EQ.DIGIT(3).AND.CARDP(2).EQ.DIGIT(2)) GO TO 70              LES01900
323 WRITE(6,123)                                                            LES01910
123 FORMAT('0',' YOUR ANSWER IS INCORRECT . VARIABLE VALUES IN THE'/' LES01920
   *LOOP ARE :'/5X,'X',5X,'Z'/5X,'1',5X,'1'/5X,'2',5X,'3'/5X,'3',5X,'6LES01930
   *'/5X,'4',5X,'10'/5X,'5',5X,'15'/5X,'6',5X,'21'/5X,'9999'//' THUS ZLES01940
   *=21'//)                                                               LES01950
 70 WRITE(6,124)                                                            LES01960
124 FORMAT('0',10X,' C. SUMMARY                                  '/5X,' LES01970
   *YOU HAVE SEEN HOW THE UNCONDITICNAL BRANCHES (''GO TO'' AND   '/' LES01980
   *''COMPUTED GO TO'') TRANSFER PROGRAM CONTROL , AND HOW THE    '/' LES01990
   *CONDITIONAL BRANCH (''IF/THEN'') TESTS FOR TRANSFER OF PROGRAM'/' LES02000
   *CONTROL ; AND YOU HAVE OBSERVED THE CONTROL OF LOOPS SO THAT  '/' LES02010
   *A PROGRAM SEGMENT MAY BE REPEATED UNTIL A SPECIFIED CONDITION '/' LES02020
   *IS MET. IN THE NEXT LESSON YOU WILL LEARN A BASIC STATEMENT TO'/' LES02030
   *CONTROL A LOOP BY THE INCREMENT AND TEST METHOD .'//' THE FOLLOWINLES02040
```

```
     *G PROBLEMS WILL TEST YOUR NEW SKILLS :'//)
      READ(5,101,END=325)                                                LES02060
  325 WRITE(6,125)                                                       LES02070
  125 FORMAT('0', 5X,' 1.) WRITE A PROGRAM TO COUNT THE NUMBERS     '/'
     *BETWEEN 50 AND 60 , AND ALSO PRINT THEM OUT . THE INPUT DATA  '/' LES02090
     *IS 10,50,35,75,62,60,54,54 .'// 5X,' 2.) WRITE A PROGRAM TO COMPUT
     *E THE PRESENT WORTH OF AN INVESTMENT       '/' FOR SOME YEARS HENCE
     * , AT VARING INTEREST RATES . THE FORMULA IS:'/5X,' P=S(1/((1+I)**LES02120
     *N))'/' WHERE P=PRESENT WORTH,S=PRINCIPAL,I=INTEREST,AND N=NR.OF YELES02130
     *ARS.'/' FOR DATA USE S=5000,AND I=.04-->.8, IN INCREMENTS OF .01 ,
     * AND N=20 '/                                                       LES02150
     */' IF YOU DESIRE TO EXECUTE THESE PROGRAMS NOW'/' REPLY: YES ; OTH
     *ERWISE REPLY : NO'//)                                             LES02170
   80 READ(5,101,END=326)CARD                                           LES02180
      CALL CRUNCH(ILNGTH)                                               LES02190
      IF(CARDP(1).EQ.ASTRSK) GO TO 80                                   LES02200
      IF(CARDP(1).EQ.ALPHA(25)) GO TO 85                                LES02210
  326 CALL EXIT
   85 CALL TEST1                                                        LES02230
      WRITE(6,127)                                                      LES02240
  127 FORMAT('0',' IF YOU DESIRE TO SEE THE SOLUTION TO THE PROBLEMS'/' LES02250
     *REPLY : YES , OTHERWISE  REPLY : NO  AND GO ON TO THE NEXT LESSON'LES02260
     *//)                                                               LES02270
   90 READ(5,101,END=325)CARD                                           LES02280
      CALL CRUNCH(ILNGTH)                                               LES02290
      IF(CARDP(1).EQ.ASTRSK) GO TO 90                                   LES02300
      IF(CARDP(1).EQ.ALPHA(14)) CALL EXIT
  328 WRITE(6,128)                                                      LES02320
  128 FORMAT('0',10X,' PROBLEM 1.)'/5X,' REM  SORT AND COUNT PROBLEM'/5XLES02330
     *,' REM C=COUNTER , N=NUMBER'/5X,' PRINT''NUMBERS IN RANGE 50->60''LES02340
     *'/5X,' LET C=0'/5X,' 1 READ N'/5X,' IF N = 9999  THEN 9999'/5X,'IFLES02350
     * N GT 60 THEN 1'/5X,' IF N LT 50 THEN 1'/5X,' LET C=C+1'/5X,' PRINLES02360
     *T N'/5X,' GO TO 1'/5X,' DATA 10,50,35,75,56,62,60,54'/5X,' 9999 PRLES02370
     *INT C,''NUMBERS IN RANGE 50->60''/5X,' END'///10X,' PROBLEM 2.)'/LES02380
     *5X,' REM COMPUTE PRESENT WORTH OF AN INVESTMENT'/5X,' REM P=PRESENLES02390
     *T WORTH,S=PRINCIPAL,N=NR,YEARS,I=VARIABLE INTREST'/5X,' PRINT''INTLES02400
     *EREST'',5X,''INVESTMENT'',5X,''PRINCIPAL'',5X,''YEARS''/5X,' READLES02410
     * S,N'/5X,' LET I=.04'/5X,'1 IF I GT.08 THEN 200'/5X,' LET P=S*(1/(LES02420
     *(1+I)**N))'/5X,' PRINT I,P,S,N'/5X,' LET I=I+.01'/5X,' GO TO 1'/5XLES02430
     *,' DATA 5000,20'/5X,' 200 END'//)                                 LES02440
      CALL EXIT
      END                                                               LES02460
C
CC    THIS LESSON INTRODUCES ITERATION , SUBSCRIPTED VARIABLES , AND
CC    LISTS(VECTORS) AND TABLES(MATRICES)
C
                              COMMON                                    LES00020
     -          STACK(100), PROG(2000), CARD(80), CARDP(80), ALPHA(48),  LES00030
```

```
     -          IAPTR, INPTR, IADATA(500) , XNDATA(500) , STRING(5),      LES00040
     -          DIGIT(10), IPRITB(10), LISTST(100), ISTLST(100),          LES00050
     -          PRT(2500), NERRS, INST, NSTLST, DEBUG, DOLSGN, QUOTE,     LES00060
     -          EQUALS, PARRT, DECMAL, PLUS, CMINUS, SLASH, COMMA,        LES00070
     -          PARLFT, ASTRSK, BLANK                                     LES00080
      COMMON  INTERP,IEXERR                                               LES00090
      REAL*8 LES5/'LESON5  '/
      CALL ALOAD(LES5,N1)
      WRITE(6,100)                                                        LES00100
  100 FORMAT('0',5X,' *** LESSON 5 ***                                '/' LES00110
     *THIS LESSON WILL INTRODUCE YOU TO ITERATION , SUBSCRIPTED     '/' LES00120
     *VARIABLES , AND  LISTS (VECTORS) AND TABLES (MATRICES).   '//5X,' LES00130
     *A. ITERATION (LOOPING)                                    '/5X,' LES00140
     *IN THE LAST LESSON YOU WERE SHOWN HOW TO USE CONDITIONAL AND   '/' LES00150
     *UNCONDITIONAL BRANCHES TO CONTROL LOOPING . THE COUNT AND TEST'/' LES00160
     *ITERATIVE LOOP OCCURS SO FREQUENTLY THAT AN ABBREVIATED BASIC '/' LES00170
     *STATEMENT HAS BEEN DEVISED TO CONTROL LOOPING . THE ITERATIVE '/' LES00180
     *LOOP HAS THE FOLLOWING FORM :                                  '//' LES00190
     *FOR << SIMPLE VARIABLE >> = << EXPRESSION >> TO << EXPRESSION >>'/
     *20X,' STEP << EXPRESSION >>'/12X,' ...'/12X,'...'/12X,'...'/10X,'
     *NEXT << SIMPLE VARIABLE >>'//)
      READ(5,101,END=302)CARD                                             LES00230
  101 FORMAT(80A1)                                                        LES00240
  302 WRITE(6,102)                                                        LES00250
  102 FORMAT('0',' FOR EXAMPLE CONSIDER THIS PROGRAM SEGMENT :  '//5X,' LES00260
     *FOR I=1 TO 10 STEP 2'/5X,' LET X=X+I'/5X,' NEXT I         '//5X,' LES00270
     *THE SIMPLE VARIABLE FOLLOWING ''FOR''IS THE LOOP INDEX . WHEN '/' LES00280
     *THE FOR/NEXT PAIR IS EXECUTED , THE LOOP INDEX IS GIVEN THE    '/' LES00290
     *VALUE (INITIALIZED) OF THE FIRST EXPRESSION (I=1 IN EXAMPLE). '/' LES00300
     *THIS INDEX IS THEN TESTED TO DETERMINE WHETHER IT IS''GREATER THAN
     *'''/' THE SECOND EXPRESSION AFTER ''TO''(10 IN EXAMPLE).IF IT IS'/
     *' GREATER, CONTROL IS TRANSFERRED TO THE STATEMENT FOLLOWING   '/'
     *''NEXT''.OTHERWISE THE REMAINING STATEMENTS WITHIN THE LOOP    '/'
     *(FOR/NEXT) ARE EXECUTED SEQUENTIALLY UNTIL THE ''NEXT'' STATEMENT
     *'/' IS REACHED.'//)
      READ(5,101,END=303)CARD                                             LES00360
  303 WRITE(6,103)                                                        LES00370
  103 FORMAT('0',' WHEN THE ''NEXT'' STATEMENT IS REACHED , THE LOOP'/' LES00380
     *INDEX IS INCREASED(INCREMENTED) BY THE AMOUNT OF THE EXPRESSION'/'LES00390
     * FOLLOWING''STEP'', AND CONTROL IS TRANSFERRED BACK TO THE''FOR'''LES00400
     */' STATEMENT WHERE THE LOOP CONTINUES UNTIL THE INDEX VALUE IS'/' LES00410
     *GREATER THAN THE FINAL VALUE . FOR EXAMPLE :               '//5X,' LES00420
     *REM DEMO LOOP. SUM THE NUMBERS FROM 1-->10.'/5X,' LET C=0'/5X,' FOLES00430
     *R I=1 TO 10 STEP 1'/5X,' LET C=C+1'/5X,' NEXT I'/5X,' PRINT''SUM='LES00440
     *',C'/5X,' END'//)                                                  LES00450
      READ(5,101,END=304)CARD                                             LES00460
  304 WRITE(6,104)                                                        LES00470
  104 FORMAT('0',' YOU WILL NOTICE THAT THE SIMPLE VARIABLE FOLLOWING'/'LES00480
```

```
   * ''NEXT'' IS THE SAME AS THE SIMPLE VARIABLE FOLLOWING''FOR'','/'
   *AND THAT THE''NEXT'' STATEMENT MARKS THE END OF THE LOOP . '/5X,' LES00500
   *BECAUSE THE INCREMENT VALUE OF A LOOP IS COMMONLY ONE(1) , THE'/' LES00510
   *''STEP'' MODIFIER AND ITS EXPRESSION MAY BE OMITTED , AND THE '/' LES00520
   * INCREMENT VALUE WILL BE ASSUMED TO BE ONE(+1). FOR EXAMPLE THE'/'LES00530
   * ABOVE ''FOR'' STATEMENT COULD BE WRITTEN :                '//5X,' LES00540
   *FOR I=1 TO 10'//5X,' THE INCREMENT VALUE AFTER''STEP'' MAY BE '/' LES00550
   *POSITIVE OR NEGATIVE ALLOWING THE FLEXIBILITY OF LOOPING FORWARD'/LES00560
   *' OR BACKWARD . FOR A NEGATIVE''STEP''VALUE THE TEST BECOMES  '/' LES00570
   *''LESS THAN''.  FOR EXAMPLE THE FOLLOWING''FOR''STATEMENTS ARE'/' LES00580
   *EQUIVALENT :'//5X,' FOR I=1 TO 10'/5X,' FOR I=10 TO 1 STEP -1'//) LES00590
    READ(5,101,END=305)CARD                                          LES00600
305 WRITE(6,105)                                                     LES00610
105 FORMAT('0',10X,' ANOTHER USEFUL TECHNIQUE OF LOOPING IS ''NESTING'LES00620
   *''/' NESTING REFERS TO PLACING ONE LOOP INSIDE ANOTHER LOOP . '/' LES00630
   *THE INNER LOOP ''SPINS''AROUND AS MANY TIMES AS THE OUTER LOOP'/' LES00640
   *IS INCREMENTED . FOR EXAMPLE CONSIDER THIS PROGRAM SEGMENT'//5X,' LES00650
   *FOR I=1 TO 10'/5X,' FOR J=1 TO 20 STEP 2'/6X,' ...'/6X,' ...'/5X,'LES00660
   * NEXT J'/6X,'NEXT I'//' IN THIS EXAMPLE THE OUTSIDE LOOP(I) IS'/'
   *REPEATED 10 TIMES,AND THE INNER LOOP(J) WOULD BE REPEATED 20  '/'
   *TIMES FOR EACH INCREMENT OF THE OUTSIDE LOOP, OR 200 REPETITIONS' LES00690
   */' LOOPS MAY BE NESTED UP TO A MAXIMUM OF 20 ; HOWEVER , THEY '/' LES00700
   *CANNOT OVERLAP . THE INNERMOST LOOP MUST BE CLOSED WITH ITS   '/' LES00710
   *''NEXT'' STATEMENT BEFORE ENCOUNTERING THE NEXT OUTER LOOP''S '/' LES00720
   *''NEXT'' STATEMENT . FOR EXAMPLE :                        '//5X,' LES00730
   *FOR X=10 TO 1 STEP -1'/5X,' FOR Y=3 TO 5'/5X,' FOR Z=-3 TO -5 STEPLES00740
   * -1'/6X,' ...'/6X,' ...'/5X,' NEXT Z'/5X,' NEXT Y'/5X,' NEXT X'//)LES00750
    READ(5,101,END=306)CARD                                          LES00760
306 WRITE(6,106)                                                     LES00770
106 FORMAT('0',10X,' WITHIN A FOR/NEXT LOOP CONDITIONAL AND UNCONDITIOLES00780
   *NAL'/'  BRANCHES MAY BE USED TO TRANSFER CONTROL OUT OF A LOOP'/'
   *OR WITHIN LIMITS OF THE SAME LOOP . HOWEVER , IT IS NOT POSSIBLE'/LES00800
   *' TO BRANCH INTO THE MIDDLE OF A FOR/NEXT LOOP  BECAUSE LOGIC '/' LES00810
   *PROBLEMS OCCUR AND AN ERROR WILL RESULT . AN ADDITIONAL ITEM  '/' LES00820
   *TO BE CAREFUL ABOUT IS USING THE INDEX VARIABLE OF THE FOR/NEXT'/'LES00830
   * LOOP IN COMPUTATIONS . IF YOU ALTER THE VALUE OF THE LOOP INDEX'/LES00840
   *' YOU WILL EFFECT THE ACTION OF THE LOOP . FOR EXAMPLE :  '//5X,' LES00850
   *FOR I=1 TO 10'/5X,' LET I=I+10'/5X,' NEXT I'/5X,' PRINT I'/5X,' ENLES00860
   *D'//' REPLY WITH VALUE OF I THAT IS PRINTED '//)                  LES00870
  5 READ(5,101,END=307)CARD                                          LES00880
    CALL CRUNCH(ILNGTH)                                              LES00890
    IF(CARDP(1).EQ.ASTRSK) GO TO 5                                   LES00900
    IF(CARDP(1).EQ.DIGIT(2).AND.CARDP(2).EQ.DIGIT(2)) GO TO 10
307 WRITE(6,107)                                                     LES00920
107 FORMAT('0',' YOUR ANSWER IS INCORRECT . THE LOOP INDEX(I) IS  '/' LES00930
   *ALTERED IN THE LOOP (I=11), AND THEN INCREMENTED AND TESTED (I=12)LES00940
   *'/' THE FINAL VALUE OF I=12.'//)                                 LES00950
 10 WRITE(6,108)                                                     LES00960
```

```
108 FORMAT('0',' YOU WILL NOW BE ASKED SOME QUESTIONS ABOUT WHAT YOU HLES00970
   *AVE JUST LEARNED .'//)                                              LES00980
    WRITE(6,109)                                                        LES00990
109 FORMAT('0',10X,' 1.) SAMPLE PROGRAM'/15X,' LET S=0'/15X,' FOR K=-5LES01000
   * TO 5'/15X,' LET S=S+K'/15X,' NEXT K'/15X,' PRINT''SUM='',S'/15X,'LES010
   * END'/' REPLY WITH VALUE OF S .'//)                                 LES01020
 15 READ(5,101,END=310)CARD                                             LES01030
    CALL CRUNCH(ILNGTH)                                                 LES01040
    IF(CARDP(1).EQ.ASTRSK) GO TO 15                                     LES01050
    IF(CARDP(1).EQ.DIGIT(1).AND.CARDP(2).EQ.BLANK) GO TO 20             LES01060
310 WRITE(6,110)                                                        LES01070
110 FORMAT('0',' YOUR ANSWER IS INCORRECT . THE VALUES OF K AND S '/' LES01080
   *IN THE LOOP ARE :'//5X,' K',5X,' S'/5X,'-5',5X,'-5'/5X,'-4',5X,'-9LES01090
   *'/5X,'-3',5X,'-12'/5X,'-2',5X,'-14'/5X,'-1',5X,'-15'/5X,' 0',5X,'-LES01100
   *15'/5X,' 1',5X,'-14'/5X,' 2',5X,'-12'/5X,' 3',5X,'-9'/5X,' 4',5X,'LES01110
   *-5'/5X,' 5',5X,' 0'/5X,' 6'//' S=0'//)                              LES01120
 20 WRITE(6,111)                                                        LES01130
111 FORMAT('0',10X,' 2.) SAMPLE PROGRAM '/15X,' FOR I=100 TO 1 STEP 2'LES01140
   */15X,' LET Z=I**2'/15X,' LET Y= Z+10'/15X,' NEXT I'/15X,' PRINT I'LES01150
   */15X,' END'//' REPLY WITH VALUE OF I'//)                            LES01160
 25 READ(5,101,END=312)CARD                                             LES01170
    CALL CRUNCH(ILNGTH)                                                 LES01180
    IF(CARDP(1).EQ.ASTRSK) GO TO 25                                     LES01190
    IF(CARDP(1).EQ.DIGIT(2) .AND.CARDP(3).EQ.DIGIT(1)) GO TO 30         LES01200
312 WRITE(6,112)                                                        LES01210
112 FORMAT('0',' YOUR ANSWER IS INCORRECT . THE INDEX VALUE OF THE'/' LES01220
   *LOOP(I=100) IS GREATER THAN THE FINAL VALUE(1) IN THE FIRST   '/' LES01230
   *TEST . THUS I=100 . IF THE LOOP WERE DECREMENTED IN STEPS OF -2'/'LES01240
   * THEN THE LOOP WOULD BE EXECUTED AND THEN I=0 .'//)                 LES01250
 30 WRITE(6,113)                                                        LES01260
113 FORMAT('0',10X,' 3.) SAMPLE PROGRAM'/15X,' LET T=0'/15X,' FOR I=5
   * TO 1 STEP -1'/15X,' FOR J=1 TO 5'/15X,' IF I=J THEN 10'/15X,' GO LES01280
   *TO 15'/15X,' 10 LET T=T+1'/15X,' PRINT 1 '/15X,' 15 NEXT J'/15X,'
   XNEXT I'/15X,' PRINT T'/15X,' END'//' REPLY WITH VALUE OF T .'//)    LES01300
 35 READ(5,101,END=314) CARD                                            LES01310
    CALL CRUNCH(ILNGTH)                                                 LES01320
    IF(CARDP(1).EQ.ASTRSK) GO TO 35                                     LES01330
    IF(CARDP(1).EQ.DIGIT(6).AND.CARDP(2).EQ.BLANK) GO TO 40             LES01340
314 WRITE(6,114)                                                        LES01350
114 FORMAT('0',' YOUR ANSWER  IS INCORRECT . THE OUTSIDE LOOP(I)  '/' L
   *IS REPEATED 5 TIMES(I=5-->1) ; AND FOR EACH VALUE OF I , THE  '/' LES01370
   *INNER LOOP(J) IS REPEATED 5 TIMES(J=1-->5) . THEREFORE  THE   '/' LES01380
   *THE CONDITIONAL BRANCH IS EXECUTED FIVE TIMES :'/5X,' I=5 , J=5'/5LES01390
   *X,' I=4 , J=4'/5X,' I=3 , J=3'/5X,' I=2 , J=2                '/5X,
   *' I=1 , J=1'//' THUS T=5.'//)                                       LES01410
 40 WRITE(6,115)                                                        LES01420
115 FORMAT('0',10X,' B. SUBSCRIPTED VARIABLES/LISTS AND TABLES'//5X,' LES01430
   *1. IN LESSON 1 YOU LEARNED THAT SIMPLE VARIABLES WERE A LETTER'/' LES01440
```

```
   *OR A LETTER FOLLOWED BY A DIGIT . SUBSCRIPTED VARIABLES CONSIST'/'LES01450
   * OF A LETTER FOLLOWED BY A SINGLE OR DOUBLE SUPSCRIPT IN PARENTHESLES01460
   *IS'/' THE SUBSCRIPTS MAY BE ANY LEGAL EXPRESSION THAT EVALUATES'/'LES01470
   * TO AN INTEGER VALUE . FOR EXAMPLE :                    '//1X,'
   *A(1) , B(3) , D(1,3) , Z(I,J) , X(3**A) , D(A(1))  ARE LEGAL SUBSCLES01490
   *RIPTED VARIABLES'//1X,' BUT  A1(1) , Z(1,2,3)  ARE ILLEGAL SUBSCRI
   *PTED VARIABLES.'//' A USE OF SUBSCRIPTED VARIABLES FOLLOWS :'//)
    READ(5,101,END=316)CARD                                          LES01520
316 WRITE(6,116)                                                     LES01530
116 FORMAT('0',5X,' 2. A NUMERIC LIST , OR VECTOR , OR SINGLE DIMENSIOLES01540
   *NED'/' ARRAY , IS A SET OF NUMERIC VALUES ARRANGED IN AN ORDERLY'/LES01550
   *' MANNER . FOR EXAMPLE , SUPPOSE YOU WOULD LIKE TO READ SOME '/'  LES01560
   *NUMBERS INTO YOUR PROGRAM AND HAVE THESE NUMBERS AVAILABLE AND'/' LES01570
   *IDENTIFIED FOR USE AT A LATER TIME . YOU COULD ASSIGN SIMPLE  '/' LES01580
   *VARIABLES TO EACH VALUE , BUT WHAT IF YOU HAD 100 NUMBERS ?   '/' LES01590
   *IT IS EASIER TO THINK OF THE NUMBERS AS A LIST OF VALUES OR A '/' LES01600
   *VECTOR THE SIZE OF WHICH IS DETERMINED BY HOW MANY NUMBERS YOU'/' LES01610
   *HAVE . YOU THEN SIMPLY ASSIGN THE VALUES TO THE LIST .'//)       LES01620
    READ(5,101,END=317)CARD                                          LES01630
317 WRITE(6,117)                                                     LES01640
117 FORMAT('0',' THE FOLLOWING PROGRAM WILL ASSIGN 10 VALUES TO THE'/'LES01650
   * LIST''A'' WHICH CONTAINS 10 ELEMENTS :'//5X,' DIM A(10)'/6X,'FOR
   *L=1 TO 10'/5X,' READ A(L)'/5X,' NEXT L'/5X,' DATA 1,3,7,5,6,9,4,2,LES01670
   *14,4'/5X,' END'//' THE LIST A , CONTAINING 10 ELEMENTS , IS REPRESLES01680
   *ENTED BY THE SUBSCRIPTED'/' VARIABLE A(L) . READ A(L) IS IN A FOR/LES01690
   *NEXT LOOP , AND'/' WITHIN THE LOOP THE LIST ELEMENTS A(1)-->A(10) LES01700
   *ARE ASSIGNED VALUES .'/' NOW THE LIST A HOLDS THE 10 VALUES AND EALES01710
   *CH VALUE IS IDENTIFIABLE .'/' CONSIDER THE FOLLOWING PROBLEM WHICHLES01720
   * SEARCHES A LIST TO FIND THE LARGEST ELEMENT :'//)
    READ(5,101,END=318)CARD                                          LES01740
318 WRITE(6,118)                                                     LES01750
118 FORMAT('0',5X,' DIM N(10)'/5X,' FOR I=1 TO 10'/5X,' LET N(I)=0'/5XLES01760
   *,' NEXT I'/5X,' READ K'/5X,' FOR J=1 TO K'/5X,' READ N(J)  '/5X,'
   *NEXT J'/5X,' LET L=N(1)'/5X,' FOR J=2 TO K'/5X,' IF N(J) LT L THEN
   * 10'/5X,' LET L=N(J)'/5X,' 10 NEXT J'/5X,' PRINT ''LARGEST NUMBER=
   *'',L'/5X,' DATA 5,3,5,9,10,6'/5X,' END'//' REPLY WITH VALUE OF L'/
   */)
 45 READ(5,101,END=319)CARD                                          LES01810
    CALL CRUNCH(ILNGTH)                                              LES01820
    IF(CARDP(1).EQ.ASTRSK) GO TO  45                                 LES01830
    IF(CARDP(1).EQ.DIGIT(2).AND.CARDP(2).EQ.DIGIT(1)) GO TO 50
319 WRITE(6,119)                                                     LES01850
119 FORMAT('0',' YOUR ANSWER IS INCORRECT . L IS ASSIGNED VALUES AS FOLES01860
   *LLOWS :'/5X,' L=3-->5-->9'//)                                    LES01870
 50 WRITE(6,120)                                                     LES01880
120 FORMAT('0',5X,' THE FIRST FOR/NEXT LOOP  ZEROS OUT THE LIST . '/' LES01890
   *IT IS A GOOD PRACTICE TO PUT SOME VALUES  IN THE LISTS YOU USE'/' LES01900
   *CTHERWISE THE COMPUTER MAY ASSIGN RANDOM VALUES . BY PUTTING  '/' LES01910
```

```
   *ZERO IN EACH ELEMENT OF THE LIST YOU ARE ASSURED THAT THE LIST'/'
   *IS CLEANED  UP BEFORE YOU USE IT.'//)                                LES01930
    READ(5,101,END=321)CARD                                             LES01940
321 WRITE(6,121)                                                        LES01950
121 FORMAT('0',5X,' 3. IN ADDITION TO LISTS(VECTORS), BASIC ALLOWS'/'
   *YOU THE ABILITY TO USE TABLES , OR MATRICES , OR TWO DIMENSIONAL'/LES01970
   *' ARRAYS . THE SUBSCRIPTS OF A TABLE REPRESENT THE ROWS AND   '/' LES01980
   *COLUMNS OF THE TABLE. THE FIRST SUBSCRIPT IS THE ROW AND THE  '/'
   *SECOND SUBSCRIPT IS THE COLUMN . FOR EXAMPLE IN TABLE''D'':'/5X,' LES01990
   *D(3,4)  REFERS TO THE VALUE OF THE ELEMENT IN ROW 3 , COLUMN 4'/' LES02000
   *OF TABLE''D''.'//)                                                LES02010
    READ(5,101,END=322)CARD                                             LES02020
322 WRITE(6,122)                                                        LES02030
122 FORMAT('0',' AS WITH A LIST , ANY ELEMENT OF A TABLE MAY BE   '/' LES02040
   *REFERENCED BY DEFINING  THE PAIR OF SUBSCRIPTS AS DESIRED IN  '/' LES02050
   *ROW-COLUMN ORDER . IN A 3X3 TABLE(MATRIX) THE TABLE IS REFERENCED'LES02060
   */' AS FOLLOWS :'//5X,' 1,1',5X,' 1,2',5X,' 1,3'/5X,' 2,1',5X,' 2,2LES02070
   X',5X,' 2,3'/5X,' 3,1',5X,' 3,2',5X,' 3,3'//' WHERE THE FIRST SUBSCLES02080
   *RIPT IDENTIFIES THE ROW AND THE SECOND THE COLUMN .          '/5X,' LES02090
   *USING THIS REFERENCE SYSTEM CONSIDER HOW TO FILL THE FOLLOWING TABLES02100
   *LE :'//5X,' 1',5X,' 2',5X,' 3'/5X,' 4',5X,' 5',5X,' 6'/5X,' 7',5X,LES02110
   *' 8',5X,' 9'//)                                                   LES02120
    READ(5,101,END=323)CARD                                             LES02130
323 WRITE(6,123)                                                        LES02140
123 FORMAT('0',' THIS TABLE''X'' COULD BE FILLED AS FOLLOWS : '//5X,' LES02150
   *DIMX(3,3)'/6X,'FOR I=1 TO  3'/6X,'FOR J=1 TO 3'/5X,' READ X(I,J)'/
   *5X,' NEXT J'/5X,' NEXT I'/5X,' DATA 1,2,3,4,5,6,7,8,9'/5X,' END'//
   *' REPLY WITH VALUE OF X(2,3)  '//)
 55 READ(5,101,END=324)CARD                                             LES02190
    CALL CRUNCH(ILNGTH)                                                 LES02200
    IF(CARDP(1).EQ.ASTRSK) GO TO 55                                     LES02210
    IF(CARDP(1) .EQ.DIGIT(7)) GO TO 60                                  LES02220
324 WRITE(6,124)                                                        LES02230
124 FORMAT('0',' YOUR ANSWER IS INCORRECT . THE VALUE OF X(2,3)=6.'/'
   *GO BACK AND LOOK AT THE REFERENCE SYSTEM . X(2,3) IS THE ELEMENT'/LES02250
   *' IN ROW 2 , COLUMN 3.'//)                                        LES02260
 60 WRITE(6,125)                                                        LES02270
125 FORMAT('0',' NOW WITH TABLE''X'' ASSIGNED VALUES CONSIDER THIS'/' LES02280
   *PROGRAM (ASSUMING TABLE''X'' HAS BEEN FILLED BY THE ABOVE PROGRAM)LES02290
   *'//5X,' LET S=0'/5X,' FOR I=1 TO 3'/5X,' FOR J=1 TO 3'/5X,' IF I NLES02300
   *E J THEN 5'/5X,' LET S=S+X(I,J)'/5X,' 5 NEXT J'/5X,' NEXT I'/5X,' LES02310
   *PRINT''SUM='',S'/5X,' END'//' REPLY WITH VALUE OF S'//)            LES02320
 65 READ(5,101,END=326)CARD                                             LES02330
    CALL CRUNCH(ILNGTH)                                                 LES02340
    IF(CARDP(1).EQ.ASTRSK) GO TO 65                                     LES02350
    IF(CARDP(1).EQ.DIGIT(2).AND.CARDP(2).EQ.DIGIT(6)) GO TO 70          LES02360
326 WRITE(6,126)                                                        LES02370
126 FORMAT('0',' YOUR ANSWER IS INCORRECT . THIS PROGRAM SUMS THE '/' LES02380
```

```
   *ELEMENTS OF THE TABLE ON THE MAJOR DIAGONAL (UPPER LEFT TO LOWER  LES02390
   *RIGHT)'/' THE SUM=15 .'//)                                       LES02400
 70 WRITE(6,127)                                                     LES02410
127 FORMAT('0',5X,' YOU HAVE SEEN HOW SUBSCRIPTED VARIABLES ARE   '/' LES02420
   *USED TO SET UP LISTS(VECTORS) AND TABLES(MATRICES). HOWEVER TO'/' LES02430
   *USE LISTS AND TABLES IN A PROGRAM , YOU MUST DIMENSION THE    '/' LES02440
   *MAXIMUM SIZE OF YOUR LIST OR TABLE SO THAT ENOUGH SPACE WILL  '/'
   *BE ALLOCATED IN THE COMPUTER MEMORY. THIS DIMENSIONING IS DONE'/'
   *WITH THE DIM STATEMENT.'//)                                       LES02470
    READ(5,101,END=328)CARD                                          LES02480
328 WRITE(6,128)                                                     LES02490
128 FORMAT('0',10X,' C. DIM STATEMENT                          '//5X,' LES02500
   *THE DIM STATEMENT TELLS THE COMPUTER THE MAXIMUM SIZE OF VECTORS'/LES02510
   *' AND TABLES THAT WILL BE USED IN YOUR PROGRAM . THE DIM STATEMENTLES02520
   *'/' MUST APPEAR IN THE PROGRAM BEFORE''ANY'' REFERENCE IS MADE'/' LES02530
   *TO THE LIST OR TABLE . IN GENERAL PRACTICE THE DIM STATEMENT   '/' LES02540
   *IS USUALLY THE FIRST STATEMENT IN THE PROGRAM . THE FORM OF THE'/'LES02550
   * DIM STATEMENT IS :'//10X,' DIM <<LIST VARIABLE>> ( <<SIZE>> )'//1
   *0X,' DIM <<TABLE VARIABLE>> ( <<SIZE,SIZE>> )'//)                LES02570
    READ(5,101,END=329)CARD                                          LES02580
329 WRITE(6,129)                                                     LES02590
129 FORMAT('0',' THE LIST AND TABLE VARIABLES ARE SUBSCRIPTED VARIABLELES02600
   *S'/' AS DEFINED EARLIER . THE SIZE IS AN UNSIGNED INTEGER IN   '/' LES02610
   *PARENTHESIS WHICH DENOTES THE MAXIMUM SIZE OF THE LIST OR TABLE'/'LES02620
   * THE DIM STATEMENT MAY CONTAIN A NUMBER OF LISTS OR TABLES     '/' LES02630
   *WITH THEIR SIZES SEPARATED BY COMMAS , FOR EXAMPLE :       '//5X,' LES02640
   *DIM A(6),X(10,3) ,Z(14,21)'//)                                   LES02650
    READ(5,101,END=330)CARD                                          LES02660
330 WRITE(6,130)                                                     LES02670
130 FORMAT('0',5X,' IF YOU TRY TO REFERENCE AN ELEMENT IN A LIST  '/' LES02680
   *OR TABLE BEYOND THE MAXIMUM SIZE IN THE DIM STATEMENT YOU WILL'/' LES02690
   *GET AN ERROR . THE MAXIMUM SIZE LIST(VECTOR) OR TABLE(MATRIX) '/' LES02700
   *ALLOWED BY THE CAI-BASIC COMPILER IN ANY ONE PROGRAM IS A TOTAL'/'LES02710
   * OF 1600 COMPUTER MEMORY SPACES .                              '/'
   *YOU WILL NOW BE ASKED SOME QUESTIONS :                         '/5X,' LES02730
   *1.) ARE THE FOLLOWING DIM STATEMENTS CORRECT,REPLY : YES OR NO'//1LES02740
   *0X,' DIM A1(10)'//)                                              LES02750
 75 READ(5,101,END=331)CARD                                          LES02760
    CALL CRUNCH(ILNGTH)                                              LES02770
    IF(CARDP(1).EQ.ASTRSK) GO TO 75                                  LES02780
    IF(CARDP(1).EQ.ALPHA(14)) GO TO 80                               LES02790
331 WRITE(6,131)                                                     LES02800
131 FORMAT('0',' THE STATEMENT IS INCORRECT . SUBSCRIPTED VARIABLES'/'LES02810
   * ARE A''SINGLE'' LETTER FOLLOWED BY ONE OR TWO SUBSCRIPTS IN '/'  LES02820
   *PARENTHESIS.'//)
 80 WRITE(6,132)                                                     LES02840
132 FORMAT('0',10X,' DIM  X(500),B(1,10),C(5,5,5)'//)                LES02850
 85 READ(5,101,END=333)CARD                                          LES02860
```

```
      CALL CRUNCH(ILNGTH)                                                LES02870
      IF(CARDP(1).EQ.ASTRSK) GO TO 85                                    LES02880
      IF(CARDP(1).EQ.ALPHA(14)) GO TO 90                                 LES02890
  333 WRITE(6,133)                                                       LES02900
  133 FORMAT('0',' THE STATEMENT IS INCORRECT . C(5,5,5) IS ILLEGAL '/' LES02910
     *CNLY SINGLE OR DOUBLE SUBSCRIPTS ARE ALLOWED.'//)                  LES02920
   90 WRITE(6,134)                                                       LES02930
  134 FORMAT('0',10X,' D. SUMMARY                                '//5X,' LES02940
     *IN THIS LESSON YOU HAVE LEARNED HOW TO USE THE FOR/NEXT STATEMENT'LES02950
     */' AND HOW TO MANIPULATE LISTS(VECTORS) AND TABLES(MATRICES)  '/' LES02960
     *BY USING SUBSCRIPTS VARIABLES AND THE DIM STATEMENT . YOU NOW '/' LES02970
     *HAVE ALL THE TOOLS TO BEGIN WRITING SOPHISTICATED PROGRAMS ;  '/' LES02980
     *AND AS YOU WRITE MORE COMPLICATED PROGRAMS , YOU WILL FIND A  '/' LES02990
     *NEED FOR SUBROUTINES . SUBROUTINES ARE COMMONLY USED PROGRAM  '/' LES03000
     *SEGMENTS THAT ARE USED OVER AGAIN IN OTHER PARTS OF YOUR PROGRAM'/LES03010
     *' SUBROUTINES ALLOW YOU TO BRANCH TO THE COMMONLY USED SEGMENT'/'
     *AND THEN RETURN TO WHERE YOU WERE AND CONTINUE EXECUTING . THE'/' LES03030
     *SUBROUTINE CAN BE''CALLED'' FROM ANYWHERE IN YOUR PROGRAM . THE'/'LES03040
     * GOSUB STATEMENT ALLOWS SUBROUTINES IN BASIC , AND YOU WILL BE'/' LES03050
     *INTRODUCED TO IT IN THE NEXT LESSON .'//)                          LES03060
      READ(5,101,END=335)CARD                                            LES03070
  335 WRITE(6,135)                                                       LES03080
  135 FORMAT('0',5X,' YOU WILL NOW BE GIVEN TWO OPTIONAL PROGRAMMING'/'  LES03090
     *PROBLEMS TO EXERCISE YOUR NEW TOOLS .'//5X,' 1.) WRITE A PROGRAM TLES03100
     *O REVERSE THE NUMBERS IN A  10-ELEMENT LIST''X'''/' IN OTHER WORDSLES03110
     * INTERCHANGE X(1) WITH X(10),X(2) WITH X(9), ETC.'/' READ IN TEN VLES03120
     *ALUES AND TEST YOUR PROGRAM BY PRINTING THE LIST'/' BEFORE AND AFTLES03130
     *ER .'//5X,' 2.) WRITE A PROGRAM TO ARRANGE THE FOLLOWING LIST IN  LES03140
     * '/' DECENDING ORDER : 10,30,5,15,40 . ONE METHOD OF APPROACHING TLES03150
     *HIS'/' IS TO CHECK THE FIRST ELEMENT OF THE LIST AGAINST THE SECONLES03160
     *D .'/' IF THE FIRST IS NOT LARGER THEN EXCHANGE THE TWO , OTHERWISLES03170
     *E'/' GO AND COMPARE THE NEXT TWO IN THE LIST. THIS PROCESS IS REPELES03180
     *ATED'/' UNTIL THERE ARE NO MORE EXCHANGES TO BE MADE . A COUNT OF LES03190
     *'/' THE EXCHANGES CAN BE MADE , AND WHEN THE COUNT EQUALS 0 THE LILES03200
     *ST IS IN ORDER.'//)                                                LES03210
      READ(5,101,END=336)CARD                                            LES03220
  336 WRITE(6,136)                                                       LES03230
  136 FORMAT('0',' IF YOU WANT TO EXECUTE THESE PROBLEMS REPLY : YES ,OTLES03240
     *HERWISE  NO ,'//)                                                 LES03250
   95 READ(5,101,END=337)CARD                                            LES03260
      CALL CRUNCH(ILNGTH)
      IF(CARDP(1).EQ.ASTRSK) GO TO 95                                    LES03270
      IF(CARDP(1).EQ.ALPHA(25)) CALL TEST1                               LES03280
  337 WRITE(6,137)                                                       LES03290
  137 FORMAT('0',5X,' IF YOU WOULD LIKE TO SEE A SOLUTION TO THE PROBLEMLES03300
     *S REPLY : YES , OTHERWISE NO ,'//)                                LES03310
   96 READ(5,101,END=338)                                                LES03320
      CALL CRUNCH(ILNGTH)                                                LES03330
```

```
      IF(CARDP(1).EQ.ASTRSK) GO TO 96                                   LES03340
      IF(CARDP(1).EQ.ALPHA(14)) CALL EXIT
  338 WRITE(6,138)                                                      LES03360
  138 FORMAT('0',' PROBLEM 1.)'//5X,' REM PROGRAM TO REVERSE ELEMENTS INLES03370
     * A LIST'/5X,' REM  Y IS A TEMPORARY LOCATION FOR THE ELEMENT BEINGLES03380
     * SWAPPED'/5X,' DIM X(10)'/5X,' FOR I=1 TO 5'/5X,' LET Y=X(I)'/5X,'LES03390
     * LET X(I)=X(11-I)'/5X,' LET X(11-I)=Y'/5X,' NEXT I'/5X,' END'//' P
     *ROBLEM 2.)'//5X,' REM  SORT LIST IN DECENDING ORDER'/5X,' REM S. ILES03410
     *S THE EXCHANGE COUNTER , Y- IS A TEMPORARY LOCATION'/5X,' DIM D(5)LES03420
     *'/5X,' FOR I=1 TO 5'/5X,' READ D(I)'/5X,' NEXT I'/5X,'5LET S=C'/5X
     *,' FOR I=1 TO N-1'/6X,'IF D(I) GE D(I+1) THEN 16'/5X,' LET Y=D(I) LES03440
     *'/6X,'LET D(I)=D(I+1)'/6X,'LET D(I+1)=Y'/6X,'LET S=1'/5X,' 16 NEXT
     * I'/5X,' IF S NE C THEN 5'/5X,' PRINT''NUMBERS IN ORDER'''/5X,' FOLES03460
     *R I=1 TO 5'/5X,' PRINT D(I) '/5X,' NEXT I'/5X,' END'//)            LES03470
      CALL EXIT
      END                                                               LES03490
C
CC    THIS LESSON INTRODUCES SUBROUTINES AND RECURSION
C
                              COMMON                                    LES00020
     -         STACK(100), PROG(2000), CARD(80), CARDP(80), ALPHA(48),  LES00030
     -         IAPTR, INPTR, IADATA(500) , XNDATA(500) , STRING(5),     LES00040
     -         DIGIT(10), IPRITB(10), LISTST(100), ISTLST(100),         LES00050
     -         PRT(2500), NERRS, INST, NSTLST, DEBUG, DOLSGN, QUOTE,    LES00060
     -         EQUALS, PARRT, DECMAL, PLUS, CMINUS, SLASH, COMMA,       LES00070
     -         PARLFT, ASTRSK, BLANK                                    LES00080
      COMMON  INTERP,IEXERR                                             LES00090
      REAL*8 LES6/'LESON6  '/
      CALL ALOAD(LES6,N1)
      WRITE(6,100)                                                      LES00100
  100 FORMAT('0',5X,' *** LESSON 6 ***                          '/'  LES00110
     *THIS LESSON WILL INTRODUCE YOU TO SUBROUTINES AND THE PROGRAMMING'LES00120
     */' TECHINQUE CALLED''RECURSION''.'//5X,' A. SUBROUTINES    '/5X,'
     *IN MANY PROGRAMS A BLOCK OF STATEMENTS MAY BE NEEDED ON MORE  '/' LES00140
     *THAN ONE OCCASION . SINCE THE REPETITION OF A BLOCK OF STATEMENTS'LES00150
     */' IS BURDENSOM , A TECHNIQUE TO ECONOMIZE ON CODING INSTRUCTIONS'LES00160
     */' CALLED A SUBROUTINE , IS PRESENTED. THE FORM OF THE SUBROUTINE LES00170
     *IS :'//5X,' GOSUB << STATEMENT NUMBER >>'/5X,' ...'/5X,' ...'/5X,'LES00180
     * ...'/5X,' RETURN'//)                                             LES00190
      READ(5,101,END=302)CARD                                           LES00200
  101 FORMAT(80A1)                                                      LES00210
  302 WRITE(6,102)                                                      LES00220
  102 FORMAT('0',5X,' EXECUTION OF THE GOSUB STATEMENT CAUSES THE   '/' LES00230
     *COMPUTER TO TRANSFER CONTROL TO THE STATEMENT NUMBER AFTER''GOSUB'LES00240
     *''/' WHEN CONTROL IS TRANSFERRED , STATEMENTS ARE EXECUTED SEQUENTLES00250
     *IALLY'/' UNTIL A ''RETURN'' IS ENCOUNTERED . AT THAT TIME , CONTROLES00260
     *L'/' IS RETURNED TO THE NEXT STATEMENT FOLLOWING THE ''GOSUB'''/' LES00270
     *STATEMENT WHICH ''CALLED'' THE SUBROUTINE . FOR EXAMPLE CONSIDER'/LES00280
```

```
   *' THE PROGRAM SEGMENT :'//)                                          LES00290
    READ(5,101,END=303)CARD                                              LES00300
303 WRITE(6,103)                                                         LES00310
103 FORMAT('0',5X,' ...'/5X,' PRINT A,B,C,D,E'/5X,' GOSUB 500'/5X,' FO
   *R I=1 TO E'/5X,' ...'/5X,' ...'/5X,' ...'/5X,' 500 REM SUBROUTINE'LES00330
   */5X,' LET X=A+B'/5X,' LET Z=B+C'/5X,' LET R=SQR(D)'/5X,' RETURN'/5LES00340
   *X,' ...'//)                                                          LES00350
    READ(5,101,END=304)CARD                                              LES00360
304 WRITE(6,104)                                                         LES00370
104 FORMAT('0',5X,' THE SUBROUTINE CONSISTS OF A SEQUENCE OF BASIC'/' LES00380
   *STATEMENTS ENDING WITH A ''RETURN'' STATEMENT . THE SUBRCUTINE'/' LES00390
   *MAY ONLY BE ENTERED FROM A ''GOSUB'' STATEMENT , AND WILL ONLY'/' LES00400
   *RETURN TO ITS PROPER PLACE AFTER ENCOUNTERING A  ''RETURN'' STATEMLES00410
   *ENT .'/5X,' ALL THE VARIABLES OF THE MAIN PROGRAM ARE AVAIABLE (PA
   *SSED) '/' TO THE SUBROUTINE , AND VICE VERSA . THE VARIABLES IN TH
   *E'/' SUBROUTINE MUST BE CHOSEN CAREFULLY SO AS NOT TO ACCIDENTLY CLES00440
   *CNFLICT'/' OR ALTER VARIABLES IN THE MAIN PROGRAM .'//)               LES00450
    READ(5,101,END=305)CARD                                              LES00460
305 WRITE(6,105)                                                         LES00470
105 FORMAT('0',5X,' THE FOLLOWING PROGRAM TO COMPUTE THE GREATEST '/' LES00480
   *CCMMON DENOMINATOR(GCD) OF THREE NUMBERS BY EUCLID''S ALGORITHM'/'LES00490
   * WILL DEMONSTRATE A USE OF SUBROUTINES :'//5X,' REM FIND GCD OF A,LES00500
   *B,C'/5X,' PRINT ''A'','''B'','''C'','''GCD'''/3X,'20 READ A,B,C'/5X,'L
   * IF C=9999 THEN 9999'/5X,' LET X=A'/5X,' LET Y=B'/5X,' REM G=GCD OLES00520
   *F A,B'/5X,' GOSUB 200'/5X,' LET X=G'/5X,' LET Y=C'/5X,' REM G=GCD LES00530
   *OF G,C'/5X,' GOSUB 200'/5X,' PRINT A,B,C,G'/5X,' GO TO 20'/11X,'20
   *0 LET Q=INT(X/Y)'/14X,' LET R=X-Q*Y'/14X,' IF R=0 THEN 300'/14X,' LES00550
   *LET X=Y'/14X,' LET Y=R'/14X,' GO TO 200'/9X,' 300 LET G=Y'/9X,' RELES00560
   *TURN'/5X,' DATA 60,90,120'/5X,' DATA 38456,64872,98765'/5X,' DATA LES00570
   *0,0,9999'/' 9999 END'//5X,' OUTPUT PRODUCED :'//5X,'A',10X,'B',10XLES00580
   *,'C',10X,'GCD'/5X,'60',9X,'90',8X,'120',9X,'30'/2X,'38456',6X,'648LES00590
   *72',6X,'98765',10X,'1'//)
    READ(5,101,END=306)CARD                                              LES00610
306 WRITE(6,106)                                                         LES00620
106 FORMAT('0',5X,'  GOSUB STATEMENTS MAY BE NESTED TO LOGICALLY  '/' LES00630
   *  CALL ANOTHER GOSUB , SIMILAR TO FOR/NEXT NESTED LOOPS . YOU '/' LES00640
   *MUST ONCE MCRE BE CAREFUL ABOUT THE STATUS OF VARIABLES BECAUSE'/'LES00650
   * VARIABLES WILL BE PASSED FROM ONE SUBROUTINE TC ANOTHER . '/5X,' LES00660
   *A SUBROUTINE THAT IS NESTED SO THAT IT CALLS ITSELF IS CALLED '/' LES00670
   *RECURSION OF SUBROUTINES OR RECURSION .'//)                           LES00680
    READ(5,101,END=307)CARD                                              LES00690
307 WRITE(6,107)                                                         LES00700
107 FORMAT('0',5X,' B. RECURSION                                  '/5X,' LES00710
   *RECURSION IS A PROGRAMMING TECHNIQUE IN WHICH SUBROUTINES CAN '/'
   *CALL THEMSELVES .FOR EXAMPLE CONSIDER THIS PROGRAM WHICH FINDS '/'LES00730
   *THE FACTORIAL OF A NUMBER USING ITERATIVE METHODS AND THEN    '/' LES00740
   *USING RECURSION :'/5X,' REM FACTORIAL :  F(N)=1*2*3*....*(N-1)*N'/LES00750
   *5X,' PRINT''ITERATIVE SOLUTION'''/6X,'PRINT''N'','''F(N)'''/3X,'10
```

```
   *READ N'/5X,' IF N GT 0 THEN 20'/5X,' GO TO 50'/3X,'20 PRINT N,'/5X
   *,' GOSUB 100'/5X,' PRINT F'/5X,' GO TO 10'/8X,'100 REM ITERATIVE
   *SOLUTION'/13X,' LET F=1'/13X,' FOR I=1 TO N'/14X,'LET F=F*I'/13X,'LES00790
   * NEXT I'/10X,' REM  RETURN F=FACTORIAL(N)'/8X,'RETURN'/3X,'50 REM
   *FACTORIAL :  F(N)= IF N=0 THEN F(N)=1'/5X,' REM  OTHERWISE , F(N)=LES00810
   *N*F(N-1)'/5X,' RESTORE'/5X,' PRINT''RESURSIVE SOLUTION''/3X,'60 R
   *EAD N'/5X,' IF N GT 0 THEN 70'/5X,' GO TO 9999'/3X,'70 PRINT N,'/5
   *X,' GO SUB 200'/5X,' PRINT F'/6X,'GO TO 60'/8X,'200 REM RECURSIVE
   *SOLUTION'/13X,' IF N GT 0 THEN 210'/13X,' LET F=1'/13X,' RETURN'/8LES00850
   *X,'210 LET N=N-1'/13X,' REM RECURSIVE CALL'/13X,' GOSUB 200'/13X,'LES00860
   * LET N=N+1'/13X,' LET F=F*N'/8X,'RETURN'/5X,' DATA 2,5,8,6,-1'/'9 LES00870
   *999 END'//)                                                      LES00880
    READ(5,101,END=308)CARD                                         LES00890
308 WRITE(6,108)                                                    LES00900
108 FORMAT('0',' IN THE ITERATIVE METHOD THE FACTORIAL FUNCTION , '/' LES00910
   *F(N) , WAS MULTIPLIED BY ITSELF IN A LOOP FROM I=1-->N . IN THE'/'LES00920
   * RECURSIVE SOLUTION  THE FACTORIAL FUNCTION ,F(N), IS DEFINED '/' LES00930
   *IN TERMS OF ITS FINAL VALUE . WHEN N=0 , F(N)=1 ; OTHERWISE   '/' LES00940
   *F(N)=N*(FACTORIAL OF N-1) .                                      '/5X,' LES00950
   *TO KEEP TRACK OF WHERE THE SUBROUTINE CALLS RETURN IN RECURSION'/'LES00960
   * IT HELPS TO EVISIONING A LAST-IN-FIRST-OUT (LIFO) STACK CONTAININ
   *G '/' THE ADDRESS OF THE GOSUB STATEMENTS. EVERY TIME A GOSUB '/'
   *IS ENCOUNTERED , PUT ITS ADDRESS ON TOP OF THE STACK (PUSHING '/' LES00990
   *DOWN ANYTHING PREVIOUSLY ON THE STACK). THEN EVERY TIME A RETURN'/LES01000
   *' IS ENCOUNTERED , RETURN TO THE TOP ADDRESS ON THE STACK (AND'/' LES01010
   *POP UP THE NEXT ADDRESS ON THE STACK).'//)                       LES01020
    READ(5,101,END=309)CARD                                         LES01030
309 WRITE(6,109)                                                    LES01040
109 FORMAT('0',' TO FULLY UNDERSTAND THE CONCEPT OF RECURSION YOU '/' LES01050
   *SHOULD STEP THROUGH THE FACTORIAL PROBLEM BY HAND USING THE   '/' LES01060
   *HELP OF THE STACK TO SEE HOW RECURSION WORKS . IF YOU UNDERSTAND'/LES01070
   *' THE CONCEPT OF RECURSION YOU ARE READY TO SOLVE THIS PROBLEM :'/LES01080
   */5X,' LET X=1'/5X,' GOSUB 100'/6X,'PRINT X'/5X,' GO TO 9999'/8X,'1LES01090
   *00 LET X=X+1'/13X,' IF X GT 3 THEN 150'/13X,' GOSUB 100'/8X,'150 LLES01100
   *ET X=X+1'/8X,'RETURN'/' 9999 END'//' REPLY WITH VALUE OF X'//)   LES01110
  5 READ(5,101,END=310)CARD                                         LES01120
    CALL CRUNCH(ILNGTH)                                             LES01130
    IF(CARDP(1).EQ.ASTRSK) GO TO 5                                  LES01140
    IF(CARDP(1).EQ.DIGIT(8).AND.CARDP(2).EQ.BLANK)GO TO 10          LES01150
310 WRITE(6,110)                                                    LES01160
110 FORMAT('0',' YOUR ANSWER IS INCORRECT . CONSIDER THE FOLLOWING'/' LES01170
   *TABLE OF VALUES FOR X , AND ITEMS IN THE STACK(1-FIRST GOSUB, 2-SELES01180
   *COND GOSUB) :'//6X,'X=1',10X,' STACK'/21X,'1'/5X,' X=2',10X,' STACLES01190
   *K '/21X,'2' /21X,'1' /5X,' X=3',10X,' STACK'/21X,'2' /21X,'2' /21X
   *,'1' /5X,' X=4'/5X,' X=5',10X,' STACK'/21X,'2'/21X,'1' /5X,' X=6',LES01210
   *10X,' STACK'/21X,'1' /5X,'X=7'//' YOU SHOULD STEP THROUGH THIS'/' LES01220
   *EXAMPLE UNTIL YOU UNDERSTAND THE RECURSION TECHNIQUE . HOWEVER,'/'LES01230
   * KNOWING HOW TO USE RECURSION IS NOT A REQUIREMENT FOR KNOWING'/' LES01240
```

```
   * HOW TO USE BASIC , IT IS ONLY A CLASSIC PROGRAMMING TECHNIQUE .'/LES01250
   *)                                                                  LES01260
 10 WRITE(6,111)                                                       LES01270
111 FORMAT('0',5X,' C. SUMMARY                                '/5X,' LES01280
   *SUBROUTINES AND RECURSION ARE A VALUABLE ADDITION TO YOUR REPERTOI
   *R'/' OF BASIC STATEMENTS . THEY SAVE BOTH PROGRAMMER TIME AND '/' LES01300
   *COMPUTER STORAGE SPACE . THE GOSUB/RETURN COMMAND AND THE OTHER'/'LES01310
   * TWELVE BASIC STATEMENTS THAT YOU HAVE ALREADY LEARNED AND USED'/'LES01320
   * FORM THE BASIC LANGUAGE .                                   '/' LES01330
   *THE FACILITY THAT YOU GAIN IN PROGRAMMING BY USING THE BASIC  '/' LES01340
   *LANGUAGE WILL DEPEND UPON HOW OFTEN YOU EXERCISE YOUR SKILLS.'/5X LES01350
   *,' THE LAST LESSON WILL PROVIDE YOU WITH A BRIEF SUMMARY OF THE'/'LES01360
   * BASIC LANGUAGE .'//)                                            LES01370
     READ(5,101,END=312)CARD                                         LES01380
312 WRITE(6,112)                                                     LES01390
112 FORMAT('0',5X,' THE FOLLOWING PROBLEMS WILL TEST YOUR LATEST SKILLLES01400
   *S :'//5X,' 1. COMPUTE THE NET SALARY , RETIREMENT CONTRIBUTION'/' LES01410
   *AND TAX FOR N EMPLOYEES. NET PAY=GROSS SALARY-RETIREMENT-TAX  '/' LES01420
   *USE SUBROUTINES TO CALCULATE :'/10X,' A) RETIREMENT CONTRIBUTION'/LES01430
   *15X,' IF SALARY <$800   , R=$0.0'/15X,' IF SALARY <$2500   , R=$10.'LES01440
   */15X,' IF SALARY >$2500   , R=$20,'/10X,' B) TAX'/15X,' IF SALARY <LES01450
   *$600   , T=$5  ; OTHERWISE'/15X,' T=.03(SALARY)'/' THE DATA IS:'/5XLES01460
   *,'EMPLOYEE',10X,'GROSS SALARY'/7X,'JONES',15X,'3000'/7X,'SMITH',15LES01470
   *X,'5500'/7X,'BROWN',15X,'475'/7X,'DALEY',15X,'10,500'/7X,'BERRY',1LES01480
   *5X,'4300'//5X,' 2. WRITE A RECURSIVE PROGRAM TO SUM THE NUMBERS'/'LES01490
   * FROM X TO Y . READ IN YOUR OWN DATA AND TEST YOUR PROGRAM .'//)
     READ(5,101,END=313)CARD                                         LES01510
313 WRITE(6,113)                                                     LES01520
113 FORMAT('0',' IF YOU WANT TO EXECUTE THESE PROGRAMS NOW , THEN '/'
   *REPLY : YES ,OTHERWISE NO'//)
 20 READ(5,101,END=313)CARD                                          LES01550
    CALL CRUNCH(ILNGTH)                                              LES01560
    IF(CARDP(1).EQ.ASTRSK)GO TO 20                                   LES01570
    IF(CARDP(1).EQ.ALPHA(14)) CALL EXIT
    CALL TEST1                                                       LES01590
    WRITE(6,114)                                                     LES01600
114 FORMAT('0',' IF YOU WOULD LIKE TO SEE A SOLUTION TO THE PROBLEMS'/
   *' REPLY: YES'//)
 25 READ(5,101,END=315)CARD                                          LES01630
    CALL CRUNCH(ILNGTH)                                              LES01640
    IF(CARDP(1).EQ.ASTRSK) GO TO 25                                  LES01650
    IF(CARDP(1).EQ.ALPHA(14)) CALL EXIT
315 WRITE(6,115)                                                     LES01670
115 FORMAT('0',' PROBLEM 1'//5X,' REM E$=EMPLOYEE,S=GROSS SALARY,R=RETLES01680
   *IREMENT'/5X,' REM T=TAX,P=NET PAY'/5X,' READ N'/5X,' FOR I=1 TO N'LES01690
   */9X,' READ E$,S'/9X,' GO SUB 20'/9X,' GOSUB 30'/9X,' P=S-R-T'/9X,'LES01700
   *PRINT E$,P,S,T,R'/5X,' NEXT I'/5X,' GO TO 9999'/9X,'20 REM TAX CAL
   *CULATION'/10X,' IF S LT 600 THEN 40'/13X,' LET T=0.03*S'/13X,' GO
```

```
      *TO 50'/9X,'40 LET T=5'/9X,'50 RETURN'/9X,'30 REM RETIREMENT CALCULLES01730
      *ATION'/13X,' IF S GE 2500 THEN 60'/13X,' IF S GE 800 THEN 70'/13X,LES01740
      *' LET R=0'/13X,' GO TO 80'/9X,'60 LET R=20'/13X,' GO TO 80'/9X,'70LES01750
      * LET R=10'/9X,'80 RETURN'/5X,' DATA 5'/5X,' DATA ''JONES'',3000'/5
      *X,' DATA ''SMITH'',5500'/5X,' DATA ''BROWN'',475'/5X,' DATA ''DALE
      *Y'',10500'/5X,' DATA ''BERRY'',4300'/' 9999 END'//' PROBLEM 2.'//5
      *X,' REM SUM FROM X-->Y'//6X,'PRINT''SUM FROM X-->Y'''/6X,'LET S=
      *0'/3X,'1 READ X,Y'/5X,' IF X=9999 THEN 9999'/5X,' GOSUB 10'/5X,' P
      *RINT X,Y,S'/5X,' GO TO 1'/9X,'10 REM SUM NUMBERS RECURSIVELY'/13X,LES01810
      *' IF X NE Y THEN 20'/13X,' LET S=X'/13X,' RETURN'/9X,'20 LET X=X+1LES01820
      *'/13X,' GOSUB 10'/13X,' LET Y=Y-1'/13X,' LET S=S+Y'/9X,' RETURN'/5LES01830
      *X,' DATA 3,6'/5X,' DATA 1,10 '/5X,' DATA 9999,1'/' 9999 END'//)   LES01840
       CALL EXIT
       END                                                              LES01860
C                                                                       LES01990
CC     THIS LESSON GIVES A COMPLETE SUMMARY OF BASIC LANGUAGE STATEMENTS LES02000
C                                                                       LES02010
                                  COMMON                                LES01880
     -          STACK(100), PROG(2000), CARD(80), CARDP(80), ALPHA(48),  LES01890
     -          IAPTR, INPTR, IADATA(500) , XNDATA(500) , STRING(5),     LES01900
     -          DIGIT(10), IPRITB(10), LISTST(100), ISTLST(100),         LES01910
     -          PRT(2500), NERRS, INST, NSTLST, DEBUG, DOLSGN, QUOTE,    LES01920
     -          EQUALS, PARRT, DECMAL, PLUS, CMINUS, SLASH, COMMA,       LES01930
     -          PARLFT, ASTRSK, BLANK                                    LES01940
       COMMON  INTERP,IEXERR                                            LES01950
       REAL*8 LES7/'LESON7  '/
       CALL ALOAD(LES7,N1)
       WRITE(6,100)
  100  FORMAT('0',5X,' *** LESSON 7 ***                                 '//'
     *THIS LESSON SUMMARIZES THE BASIC DEFINITIONS AND BASIC STATEMENTS'
     */' THAT YOU HAVE LEARNED IN CAI-BASIC .'//5X,' A. BASIC DEFINITION
     *S'//' 1. ALPHA-NUMERIC CHARACTERS :                             '//5X,'
     *A) DIGITS : 0-->9                                               '//5X,'
     *B) LETTERS : A-->Z                                              '//5X,'
     *C) SPECIAL CHARACTERS :  ** * / + - ( ) = '' , $                  '//'
     *2. STATEMENT NUMBERS : ONE TO FOUR DIGITS (0-->9999)             '//'
     *3. STRING : ANY SEQUENCE OF ALPHA-NUMERIC CHARACTERS ENCLOSED '/'
     *IN ''SINGLE'' QUOTES .  EG. ''HELP'' '//)
       READ(5,101,END=302)CARD
  101  FORMAT(80A1)
  302  WRITE(6,102)
  102  FORMAT('0',' 4. NUMBERS : (LIMITED TO 9 DIGITS)                 '//5X,'
     *A) INTEGERS :  DIGITS WITH NO FRACTIONAL PART . EG. 5,7,10'//5X,'
     *B) REAL : DIGIT WITH A FRACTIONAL PART . EG. 5.0,7.31,-16.0    '/'
     *A NUMBER MAY BE PRECEEDED BY A SIGN (+,-) , BUT IS ASSUMED TO '/'
     *BE POSITIVE IF NONE IS GIVEN .                                   '//'
     *5. VARIABLES :                                                 '//5X,'
     *A) SIMPLE VARIABLES : A SINGLE LETTER , OR A SINGLE LETTER     '/'
```

```
   *FOLLOWED BY A DIGIT . EG  A,A1,B6,Z0                          '//5X,'
   *B) ALPHA VARIABLES : A SINGLE LETTER FOLLOWED BY A DOLLAR SIGN ($)
   *'/5X,' EG. A$,V$                                              '//5X,'
   *C) SUBSCRIPTED VARIABLE : (SINGLE LETTER)                    '//10X,'
   *(1) SINGLE SUBSCRIPT :  <<LETTER>> ( <<EXPRESSION>> )        '//10X,'
   *(2) DOUBLE SUBSCRIPT :  <<LETTER>> ( <<EXPRESSION,EXPRESSION>> )'/
   *'/' EG. A(5),Z(5,10),X(A*B,X**2)'//)
    READ(5,101,END=303)CARD
303 WRITE(6,103)
103 FORMAT('0',' 5. OPERATORS : (LISTED IN DECENDING HIERARCHY)'//5X,'
   * A) EXPONENTIATION  **                                        '/5X,'
   *B) MULTIPLICATION  *                                         '/5X,'
   *C) DIVISION        /                                         '/5X,'
   *D) ADDITION        +                                         '/5X,'
   *E) SUBTRACTION     -                                           '//'
   *6. EXPRESSIONS :                                            '//5X,'
   *A) SINGLE NUMBER OR VARIABLE   EG. 10,-.53,D,X(I,J)          '//5X,'
   *B) BUILT IN FUNCTION   EG. SQR(10),TAN(.75)                  '//5X,'
   *C) ARITHMETIC EXPRESSION : EXPRESSIONS SEPARATED BY OPERATORS '/'
   *AND GROUPED BY PARENTHESIS . EG. 5+6.0 , A**2(3*X-4)  '//)
    READ(5,101,END=304)CARD
304 WRITE(6,104)
104 FORMAT('0',' 7. RELATIONS :                                  '//5X,'
   *A) GT    A GT B    A GREATER THAN B                          '/5X,'
   *B) GE    A GE B    A GREATER THAN OR EQUAL TO B              '/5X,'
   *C) LT    A LT B    A LESS THAN B                             '/5X,'
   *D) LE    A LE B    A LESS THAN OR EQUAL TO B                 '/5X,'
   *E) NE    A NE B    A NOT EQUAL TO B                          '/5X,'
   *F)  =    A = B     A EQUAL TO B                            '///5X,'
   *B. BASIC STATEMENTS                                           '//'
   *1. REM << ANY SET OF ALPHA-NUMERIC COMMENTS >>                '//'
   *2. READ << VARIABLE,....,VARIABLE >>                          '//'
   *3. END                                                        '//)
    READ(5,101,END=305)CARD
305 WRITE(6,105)
105 FORMAT('0',' 4. LET << VARIABLE >> = << EXPRESSION >>         '//'
   *5. PRINT << ''STRING'' OR EXPRESSION,....,''STRING'' OR EXPRESSION
   * >>'/5X,' OR SIMPLY   PRINT                                   '//'
   *6. DATA  << ''STRING'' OR NUMBER,....,''STRING'' OR NUMBER >>'//'
   *7. RESTORE  OR  RESTORE$                                     '//'
   *8. IF << EXPRESSION >> << RELATION >> THEN << STATEMENT NUMBER >>'
   *//' 9. GO TO << STATEMENT NUMBER >>                           '//'
   *10. ON << EXPRESSION >> GO TO << STATEMENT NUMBER,....,STATEMENT N
   *UMBER >>'//' 11. FOR << SIMPLE VARIABLE >> = << EXPRESSION >>'/30X
   *,' TO << EXPRESSION >> STEP << EXPRESSION >>                 '/5X,'
   *....'/5X,' ...'/5X,'...'/' NEXT << SIMPLE VARIABLE >>          '//'
   *12. DIM << LETTER >> ( << INTEGER EXPRESSION >> )       OR    '/'
   *    DIM << LETTER >> ( << INTEGER EXPRESSION,INTEGER EXPRESSION >>
```

```
     * )'//' 13. GOSUB << STATEMENT NUMBER >>                    '/5X,'
     *...'/5X,' ...'/5X,' ...'/' RETURN '//)
      READ(5,101,END=306)CARD
  306 WRITE(6,106)
  106 FORMAT('0',' THAT CONCLUDES YOUR INSTRUCTION WITH CAI-BASIC . '/'
     *HOWEVER , YOU ARE INVITED TO USE WHATEVER FACILITIES OF CAI-BASIC'
     */' YOU DESIRE AT ANY TIME . IF YOU HAVE NOT RUN PROGRAMS UNDER'/'
     *THE OS/BATCH MODE (PUNCHING YOUR OWN CARDS AND HANDING THEM   '/'
     *ACROSS THE COUNTER TO BE RUN), YOU SHOULD GET THE BASIC MANUAL,'/'
     * TECHNICAL NOTE NR. 0211-12 , IN ROOM I-147 TO FIND THE PROPER'/'
     *JCB CONTROL CARDS REQUIRED . GOOD LUCK WITH THE COMPUTER , AND'/'
     *REMEMBER IT ONLY DOES WHAT YOU TELL IT TO DO - GIGO (GARBAGE IN'/'
     * GARBAGE OUT ).'//)
      CALL EXIT
      END                                                        LES02030
```

APPENDIX C

```
13.13.46 START
EXECUTION BEGINS...

 HI , WELCOME TO CAI-BASIC . THERE ARE ONLY A FEW
SIMPLE RULES TO REMEMBER IN ORDER TO HAVE A SUCCESSFUL SESSION
ON THE TERMINAL WITH CAI-BASIC :

     1. WHEN ASKED FOR A RESPONSE , TYPE IN THE CORRECT REPLY AND
HIT THE CARRIAGE RETURN KEY ON THE RIGHT SIDE OF THE KEYBOARD.

     2. IF YOU MAKE A TYPING ERROR WHILE MAKING ANY RESPONSE
OR INPUT , TYPE IN FOUR DOLLAR SIGNS ($$$$) AFTER THE ERROR OR
ANYWHERE ON THAT INPUT LINE AND HIT CARRIAGE RETURN .THE
ENTIRE LINE WILL THEN BE IGNORED AND YOU CAN TYPE IN THE CORRECT
INPUT OR RESPONSE .

     3. IF AT ANY POINT IN THE SESSION YOU WANT TO STOP THE SESSION
TYPE IN THE WORD 'QUIT' AS SOON AS YOU ARE ASKED FOR THE NEXT
RESPONSE ,HIT CARRIAGE RETURN,THEN HIT ATTN KEY AND TYPE LOGOUT .

     4. DURING YOUR TERMINAL SESSION CAI-BASIC WILL HALT OCCASIONALLY
TO LET YOU READ A SEQUENCE OF INFORMATION . WHEN YOU ARE
READY TO CONTINUE , TYPE IN 'GO' , AND HIT CARRIAGE RETURN

     5. DURING YOUR TERMINAL SESSION YOU MAY NOTICE THAT
THE TYPING IS NOT ALWAYS PERFECT . SOME DAYS THE COMPUTER IS
NOT UP TO PAR AND YOU WILL HAVE TO ADJUST TO THE MINOR IRRITANT

  IF YOU ONLY WANT TO EXECUTE PROGRAMS AT THIS TIME THEN
REPLY : YES  ; OTHERWISE REPLY : NO   .


no

 IF THIS IS YOUR FIRST SESSION WITH CAIBASIC, THEN
REPLY:  YES  ;OTHERWISE REPLY:  NO  .


yes

          CAIBASIC IS A PROGRAM TO TEACH YOU THE
FUNDAMENTALS OF A PROGRAMMING LANGUAGE.THE LANGUAGE TO BE LEARNED
IS BASIC ; A SIMPLE LANGUAGE FOR THE USER WHO HAS LITTLE
KNOWLEDGE OF COMPUTERS AND WHOSE PRIMARY INTEREST IS IN OBTAINING
ESULTS.
          THE SIMPLICITY OF THE BASIC LANGUAGE AND ITS RANGE
OF CAPABILITIES SHOULD ALLOW YOU TO LEARN THE LANGUAGE AND
```

WRITE PROGRAMS IN A MINIMAL AMOUNT OF TIME.

THE REFERENCE TEXTS RECOMMENDED FOR CAIBASIC ARE :

1. BASIC LANGUAGE MANUAL , TN # 0211-12 APRIL 1971
(FREE UPON REQUEST IN I-147)

2. INTRODUCTION TO COMPUTING THROUGH THE BASIC LANGUAGE , R.L. NOLAN
(BOOKSTORE / MAIN LIBRARY)

3. BASIC PROGRAMMING , V.C. HARE
(COMPUTER CENTER LIBRARY)

*** AFTER YOU HAVE FINISHED READING AN INPUT , TYPE IN GO AND HIT THE RETURN AND THE PROGRAM WILL CONTINUE ***

go

DURING YOUR TERMINAL SESSION YOU WILL BE LEARNING THE STRUCTURE OF THE LANGUAGE BASIC. THE INSTRUCTION SET CONTAINS 7 LESSONS AND YOU MAY PROCEED THROUGH THE LESSONS AT YOUR OWN SPEED.

THE LESSONS ARE AS FOLLOWS :

LESSON 1 PROGRAM FORMAT AND BASIC DEFINITIONS
LESSON 2 REMARKS,INPUT/OUTPUT AND DATA
LESSON 3 ASSIGNMENT STATEMENTS AND BUILT IN FUNCTIONS
LESSON 4 BRANCHING
LESSON 5 LOOPING AND SUBSCRIPTED VARIABLES
LESSON 6 SUBROUTINES AND RECURSION
LESSON 7 SUMMARY OF BASIC STATEMENTS

** WHEN YOU ARE READY TO CONTINUE , TYPE IN GO AND HIT THE CARRIAGE RETURN ***

go

IN EACH LESSON YOU WILL BE GIVEN INSTRUCTION SEQUENCES AND THEN YOU WILL BE ASKED QUESTIONS TO SEE IF YOU UNDERSTOOD THE INSTRUCTIONS.THE QUESTIONS WILL BE OF VARIOUS TYPES: MULTIPLE CHOICE,TRUE/FALSE,ACTUAL PROGRAM STATEMENTS,ETC. YOU WILL BE PROMPTED FOR YOUR ANSWER, AND WHEN READY TYPE IN OUR RESPONSE AND HIT THE RETURN KEY.

IF YOU KEEP THE TELETYPE OUTPUT FROM YOUR TERMINAL SESSION YOU WILL HAVE A READY REFERENCE FOR FUTURE USE

```
go

      *** LESSON 1. ***

THIS INSTRUCTION SET WILL INTRODUCE YOU TO THE STRUCTURE OF BASIC
LANGUAGE STATEMENTS , THE RULES FOR VARIABLES AND NUMBERS , AND
THE SYMBOLS FOR ARITHMETIC OPERATIONS.

***DONT FORGET TO TYPE GO AND HIT RETURN WHEN READY TO CONTINUE . ***


go
      A.  PROGRAM STRUCTURE
                THE BASIC LANGUAGE , AND ALL OTHER LANGUAGES ,
HAS A SPECIFIED STRUCTURE . EACH BASIC STATEMENT HAS A REQUIRED
FORM WITH POSSIBLY ONE OR MORE VARIATIONS. THE FOLLOWING
EXAMPLE WILL ILLUSTRATE SOME SIMPLE BASIC STATEMENTS IN THE
PROPER PROGRAM STRUCTURE

          REM  PROGRAM TO COMPUTE GAS MILAGE
          REM  M=MILES TRAVELED , G=GAS USED
           READ M,G
           LET T=M / G
           PRINT 'MILES TRAVELED','GAS USED',' MILES/GAL'
           PRINT M,G,T
           DATA 500,25
           END
     OUTPUT PRODUCED IS :
MILES TRAVELED  GAS USED    MILES/GAL
      500           25          20


go

           AS YOU CAN SEE FROM THE ABOVE SAMPLE, THE PROGRAM
STRUCTURE CONSISTS OF BASIC LANGUAGE STATEMENTS FOLLOWED
BY AN END STATEMENT . THE WORDS : REM , READ, LET ,  PRINT , DATA
AND END   ARE KEY WORDS THAT MAKE UP A BASIC STATEMENT .


go

           MOST OF THE KEY WORDS USED IN THE BASIC
STATEMENTS ARE SELF-EXPLANATORY:

     REM  ALLOWS REMARKS/COMMENTS
     READ M,G  ASSIGNS NUMBERS IN THE DATA STATEMENT TO THE
               VARIABLES M AND G
     LET  ASSIGNS THE RESULT OF M DIVIDED BY G INTO VARIABLE T
```

```
     PRINT'STRING'  CAUSES THE STRING IN SINGLE QUOTES TO BE
                      PRINTED LITERALLY
     PRINT M,G,T  PRINTS THE CURRENT VALUE OF THE VARIABLES M,G,T
     END  TELLS THE COMPUTER THAT THE INPUT PROGRAM IS TO BE
            EXECUTED
```

go

```
          IF AT THIS POINT YOU WOULD LIKE TO RUN THE
SAMPLE PROGRAM TO GAIN SOME CONFIDENCE IN THE COMPUTER AND ITS
ABILITY TO PROVIDE SPEEDY RESULTS, THEN REPLY : YES   ; OTHER-
WISE REPLY : NO  AND THE INSTRUCTION WILL CONTINUE .
```

no

```
     B. PROGRAM FORMAT

     A BASIC PROGRAM CONSISTS OF A SEQUENCE OF BASIC STATEMENTS , ONE
STATEMENT PER INPUT LINE , FOLLOWED BY AN END STATEMENT .
BECAUSE NO BASIC STATEMENT MAY BE LONGER THAN ONE INPUT LINE (80 SPACES) ,
THERE IS NO PROVISION FOR CONTINUING STATEMENTS FROM ONE LINE
TO THE NEXT . HOWEVER ; YOU MAY SPACE THE INPUT LINE
AS DESIRED FOR READABILITY SINCE THE COMPUTER IGNORES BLANKS IN BASIC .
```

go

```
          1. STATEMENT NUMBERS
EACH BASIC STATEMENT MAY HAVE AN OPTIONAL
STATEMENT NUMBER PRECEDING IT FOR IDENTIFICATION PURPOSES .
THIS STATEMENT NUMBER MUST BE AN INTEGER BETWEEN 1 -> 9999
     FOR EXAMPLE :    12 READ M,G

          2. KEY WORDS
THE KEY WORDS THAT MAKE UP A BASIC STATEMENT (REM ,READ ,LET ,ETC,)
ARE SPECIAL TERMINAL SYMBOLS THAT ARE RECOGNIZED BY THE COMPUTER
AND FOR THIS REASON THEY MUST BE SPELLED CORRECTLY AND ONLY
USED IN BASIC STATEMENTS .
          THE END STATEMENT INDICATES THAT THE INPUT PROGRAM IS
COMPLETED AND THAT PROGRAM EXECUTION IS TO BEGIN . THE 'END'
STATEMENT IS ALWAYS THE LAST STATEMENT IN A PROGRAM .
```

go

```
 YOU WILL NOW BE ASKED A FEW SIMPLE QUESTIONS ABOUT
WHAT YOU HAVE JUST LEARNED .


          1.) IS THIS A LEGAL BASIC PROGRAM(REPLY : YES OR NO)??
```

```
     REM ONE LINE DO NOTHING PROGRAM
     END


yes

 YES , THE SIMPLEST BASIC PROGRAM CONSISTS OF JUST AN  'END' STATEMENT.

          2.) WHICH OF THE FOLLOWING BASIC STATEMENTS IS
IN THE PROPER FORMAT :
               A.) 15 LET T=M/G
               B.) 15LETT=M/G
               C.) 15  L E T   T = M / G

          REPLY A , B , C  OR ALL .

all

 ALL OF THE ABOVE BASIC STATEMENTS ARE CORRECT BECAUSE
SPACES ARE DISREGARDED . NOTE THAT STATEMENT B.) , ALTHOUGH
CORRECT IS CONFUSING TO READ . BLANKS AND INDENTATIONS MAKE A
PROGRAM EASY TO READ .


      C. ALPHA-NUMERIC CHARACTERS

     ALPHA-NUMERIC CHARACTERS ARE THE LEGAL CHARACTERS , DIGITS , AND
SPECIAL CHARACTERS THAT CAN BE USED IN BASIC .
          1. CHARACTERS
CHARACTERS CONSIST OF THE LETTERS IN THE ALPHABET  A-->Z

          2. DIGITS
DIGITS ARE THE SINGLE NUMBERS  0-->9

          3. SPECIAL CHARACTERS
 THE SPECIAL CHARACTERS ARE  **  *  /  +  -  (  )  =  '  , $

          4. STRING
A STRING IS ANY LIST OF ALPHA-NUMERIC CHARACTERS
ENCLOSED IN 'SINGLE' QUOTES . FOR EXAMPLE :
     'THIS IS A STRING'


go
```

D. VARIABLES

IN BASIC THERE ARE THREE TYPES OF VARIABLES : SIMPLE , ALPHA AND SUBSCRIPTED . SUBSCRIPTED VARIABLES WILL BE COVERED IN LESSON 5 .

1. SIMPLE VARIABLES

SIMPLE VARIABLES ARE IDENTIFIED BY A SINGLE LETTER OR A SINGLE LETTER FOLLOWED BY A DIGIT BETWEEN 0->9 .

FOR EXAMPLE : A , A3 , Z0 , AND X ARE LEGAL VARIABLES ; BUT A26 , 9Z , ABC , AND X1Y ARE ILLEGAL VARIABLES . THEREFORE ; YOU HAVE 286 SIMPLE VARIABLES FOR USE IN YOUR PROGRAMS.

go

2. ALPHA VARIABLES

ALPHA VARIABLES ARE USED FOR ALPHA-NUMERIC MANIPULATIONS IN WHICH A GROUP OF ALPHA-NUMERIC CHARACTERS , CALLED A STRING , ARE REPRESENTATED BY AN ALPHA VARIABLE . THE ALPHA VARIABLE CONSISTS OF A SINGLE LETTER FOLLOWED BY A DOLLAR SIGN , $. THE MAXIMUM LENGTH OF THE ALPHA-NUMERIC STRING ASSIGNED TO THE ALPHA VARIABLE IS 16 CHARACTERS.

FOR EXAMPLE : A$, X$, Z$ ARE LEGAL ALPHA VARIABLES.

A SIMPLE USE OF ALPHA VARIABLES FOLLOWS :

```
READ A$,B$
PRINT A$,B$
DATA 'MONDAY','21 JUNE'
END
```

THIS PROGRAM WILL PRODUCE THE OUTPUT :

```
MONDAY 21 JUNE
```

go

E.NUMBERS

NUMBERS MAY BE EXPRESSED AS INTEGERS OR AS REALS ,AND A NUMBER WHETHER INTEGER OR REAL IS LIMITED TO 9 DIGITS , NOT INCLUDING DECIMAL POINT. A NUMBER IS ASSUMED POSITIVE UNLESS IT IS PRECEEDED BY A - SIGN .

1. INTEGERS

INTEGERS ARE NUMBERS WITH NO FRACTIONAL PART ,IE. 3 , 15

2. REALS

REAL (FIXED-POINT) NUMBERS HAVE A DECIMAL POINT AND A FRACTIONAL PART ,IE. 3.0 , 15.31, 729.1 , 0.0

go

F. EXPRESSIONS

AN EXPRESSION MAY BE A SINGLE CONSTANT OR VARIABLE , OR AN ARITHMETIC EXPRESSION . ARITHMETIC EXPRESSIONS ARE FORMED BY USING OPERATORS AND PARENTHESIS .

1. OPERATORS

NUMBERS AND SIMPLE VARIABLES MAY BE COMBINED INTO ARITHMETIC EXPRESSIONS BY USING ONE OR MORE OF THE ARITHMETIC OPERATORS :

**	EXPONENTIATION
*	MULTIPLICATION
/	DIVISION
+	ADDITION
-	SUBTRACTION

IN WRITING AN EXPRESSION , EACH ARITHMETIC OPERATION MUST BE SHOWN. NO TWO OPERATIONS MAY BE ADJACENT , NOR MAY TWO NUMBERS OR VARIABLES BE ADJACENT IN AN EXPRESSION.
FOR EXAMPLE: 2+2 , A/B, B+C , X*Z , AND Y-3.0 ARE LEGAL EXPRESSIONS.

go

2. PARENTHESIS

PARENTHESIS MAY BE USED TO GROUP EXPRESSIONS AND TO CONTROL THE ORDER IN WHICH AN ARITHMETIC EXPRESSION IS EVALUATED . THE NORMAL HIERARCHY OF OPERATORS IS :

1. **
2. * AND /
3. + AND -

IF PARENTHESIS ARE USED , THE EXPRESSION WITHIN A PARENTHESIS PAIR IS EVALUATED FIRST . PARENTHESIS IN EXPRESSIONS MUST ALWAYS OCCUR IN PAIRS ,IE. 5 * (4 + 3) , (5*(4+3)) ARE EQUIVALENT EXPRESSIONS HAVING THE VALUE 35 .

OPERATIONS ARE EVALUATED ACCORDING TO THE OPERATORS HIERARCHY IF TWO OPERATORS OF THE SAME HIERARCHY OCCUR IN AN EXPRESSION EVALUATION IS FROM LEFT TO RIGHT .
FOR EXAMPLE: 2*4 + 6/3

IN THIS EXPRESSION 2*4 WILL BE EVALUATED FIRST , THEN 6/3 , AND THEN THE TWO RESULTS WILL BE ADDED .

```
go

 YOU WILL NOW BE ASKED A FEW SIMPLE QUESTIONS ABOUT
WHAT YOU HAVE JUST LEARNED .


          1.) 4 + 6 / 2  HAS THE VALUE (REPLY WITH VALUE)

5

 YOUR ANSWER IS INCORRECT . THE EXPRESSION IS EVALUATED AS FOLLOWS :
          4+6/2  ->  4+3  ->  7

          2.) (4+6)/2  HAS THE VALUE (REPLY WITH VALUE)

5

          3.) ((4+6)/2)**2  HAS THE VALUE (REPLY : VALUE)

25

       F. SUMMARY
                THIS CONCLUDES THE INSTRUCTION SET FOR LESSON 1.
YOU WILL NOW BE GIVEN A SAMPLE BASIC PROGRAM AND ASKED TO FIND
THE MISTAKES IN IT , CONCERNING WHAT YOU HAVE LEARNED IN THIS
LESSON :

     1  REM REVIEW PROGRAM
     2  READ A1,B2,Z1,$Y
     3  LET A1=B2(A1+Z1)
     4  LET B2=((A1*3)+Z1**2)
     5  LET Z1=3.141592763
     6  PRINT 1A,B2,Z1
     7  DATA 5.0,3,3.1416,'FINI'

AFTER LOOKING AT THE SAMPLE PROBLEM , YOU WILL BE ASKED QUESTIONS
ABOUT EACH STATEMENT ,

go

          1.)  LINE 2  CONTAINS A READ STATEMENT FOLLOWED
BY A LIST OF VARIABLES . ANY ERRORS (REPLY: YES OR NO)?

yes
```

```
 INPUT CORRECTION TO THE INCORRECT VARIABLE ONLY .

y$

          2.)  LINE 3  CONTAINS A LET STATEMENT FOLLOWED
BY A1 = EXPRESSION . ANY ERRORS (REPLY : YES OR NO) ?


yes

 INPUT CORRECTION TO ILLEGAL EXPRESSION ; EVERYTHING
AFTER = SIGN .


b2*(a1+z1)

          3.)  LINE 4  IS SIMILAR TO LINE 3 . ANY ERRORS
REPLY   YES OR NO) ?


no

          4.)  LINE 5  CONTAINS A LET STATEMENT FOLLOWED
BY A NUMBER . ANY ERRORS (REPLY : YES OR NO) ?


no

 THE NUMBER CONTAINS 10 DIGITS AND THE MAXIMUM ALLOWED
IS 9 DIGITS.

          5.)  LINE 6  CONTAINS A PRINT STATEMENT FOLLOWED
BY A LIST OF VARIABLES . ANY ERRORS (REPLY : YES OR NO)?


yes

 INPUT CORRECTION TO THE INCORRECT VARIABLE ONLY .

a1

          THERE ARE NO ERRORS IN LINE 7 . IS THE PROGRAM
READY TO EXECUTE (ASSUMING ABOVE ERRORS CORRECTED)
(REPLY : YES OR NO)


yes

          THE PROGRAM FORMAT REQUIRES THAT AN END STATEMENT
```

```
BE THE LAST STATEMENT OF THE PROGRAM . THUS THIS SAMPLE PROGRAM
WOULD NOT EXECUTE .

THIS CONCLUDES THE REVIEW QUESTIONS
FROM LESSON 1 .


          DO YOU WANT TO TERMINATE YOUR INSTRUCTION SESSION ?
 (REPLY : YES OR NO )

no

          IF YOU WANT TO EXECUTE PROGRAMS AT THIS TIME
THEN REPLY : YES ; OTHERWISE REPLY : NO AND THE INSTRUCTION
WILL CONTINUE .

no
```

```
EXECUTION BEGINS...

      *** LESSON 2. ***

THIS INSTRUCTION SEQUENCE WILL INTRODUCE YOU TO THE BASIC LANGUAGE
STATEMENTS  REM , PRINT , READ , AND DATA . USING THESE STATEMENTS
YOU WILL BE ABLE TO CONSTRUCT AND EXECUTE ELEMENTARY PROGRAMS .
 THE FORM FOR EACH BASIC STATEMENT WILL INCLUDE :

         'KEY WORD' << ELEMENTS OR LIST OF ELEMENTS SEPARATED BY COMMAS >>

THE CARET SYMBOLS '<< >>' DELINIATE THE LEGAL ITEMS THAT MAY
FOLLOW THE KEYWORD AND MAKE UP THE BASIC STATEMENT .


go

           A. REM
     THE BASIC STATEMENT WHICH ALLOWS YOU TO INSERT
REMARKS INTO YOUR PROGRAM IS IDENTIFIED BY THE KEY WORD 'REM'.
REM IS A NON-EXECUTING STATEMENT WHICH MAY BE USED OPTIONALLY
AT ANY PLACE IN YOUR PROGRAM TO INTRODUCE A PROGRAM NAME , TO
EXPLAIN VARIABLES, TO DOCUMENT YOUR PROGRAM , ETC.


go

      THE FORM FOR THE REM STATEMENT IS :

                REM << ANY SEQUENCE OF ALPHA-NUMERIC CHARACTERS >>

THE REM STATEMENT IS IGNORED BY THE CAIBASIC COMPILER , AND IS
ONLY FOR YOUR INFORMATION .                    FOR EXAMPLE :

          REM  PROGRAM TO COMPUTE INCOME TAX


go

           B. PRINT
     THE PRINT STATEMENT IS THE METHOD OF WRITING OUT THE RESULTS OF
THE BASIC PROGRAM , TO DISPLAY VALUES OF VARIABLES , TO LABEL
STATEMENT IS :

 PRINT << EXPRESSION OR 'STRING',....,EXPRESSION OR 'STRING' >>

MULTIPLE ELEMENTS IN THE PRINT LIST ARE SEPARARED BY COMMAS.


go

      1. PRINT << EXPRESSION >>  WILL WRITE OUT THE
CURRENT VALUE OF THE EXPRESSION , WHERE AN EXPRESSION AS DEFINED
IN LESSON 1 WAS A NUMBER , A VARIABLE , OR AN ARITHEMTIC
```

```
EXPRESSION THAT IS TO BE EVALUATED .
          FOR EXAMPLE :

     PRINT 1,A,B1,5**2

ASSUMING THAT A=10.0 , B=13.3  WOULD PRINT :

     1          10.0           13.3          25


go

      2. PRINT <<'STRING' >>  WILL PRINT OUT ALL THE
ALPHA-NUMERIC CHARACTERS OF THE STRING WITHIN 'SINGLE' QUOTES.
THIS FORM IS USED FOR LABELING THE COMPUTER OUTPUT.

FOR EXAMPLE :   PRINT 'THE ANSWER IS :'

PRODUCES THE RESULT :

     THE ANSWER IS :

     3. PRINT   BY ITSELF IS USED TO SKIP A LINE ON THE COMPUTER OUTPUT .


go

 THE COMPUTER OUTPUT SHEET IS DIVIDED INTO 8 ZONES
EACH 15 COLUMNS WIDE . PRINT ZONES CAN BE SKIPPED BY PUTTING A
BLANK IN THE PRINT LIST . FOR EXAMPLE :
                PRINT A, B, ,X
OUTPUTS THE VALUE OF A IN THE FIRST ZONE , B IN THE SECOND ,
SKIPS THE THIRD ZONE , AND PUTS X IN THE FOURTH ZONE .
        ALPHA VARIABLES AND 'STRINGS' MAY EXTEND OVER SEVERAL
ZONES , BUT NUMERIC RESULTS ARE LEFT ADJUSTED IN THE SPECIFIED
ZONE . IF MORE THAN EIGHT ITEMS OCCUR IN THE PRINT LIST , THE
ITEMS WILL OVERFLOW AND BE PRINTED ON THE NEXT LINE .


go

 USING THE PRINT AND END STATEMENT YOU NOW HAVE THE
FACILITY TO EXECUTE YOUR FIRST PROGRAMS . FOR EXAMPLE :
          REM  PROGRAM TO COMPUTE THE SQUARE OF A NUMBER
          PRINT'5 SQUARED =',5**2
          END

PRODUCES THE RESULT :

     5 SQUARED =  25

YOU WILL NOW BE GIVEN TWO PROBLEMS TO SOLVE . YOU WILL ENTER
```

THE EXECUTION PHASE OF CAIBASIC WHERE YOU CAN RUN YOUR PROBLEMS
AND THEN YOU WILL RETURN TO FINISH THE LESSON.

1) FIND THE SQUARE ROOT OF 5 SQUARED MINUS 4 TIMES 2 TIMES 2 .

2) FIND THE VALUE OF 3.1416(B CUBED)H/12
WHERE B=2.50 , H=3.03 .

IF YOU WISH TO SKIP THESE PROBLEMS REPLY : YES
AND THE LESSON WILL CONTINUE.

yes

C. READ

THE READ STATEMENT IS THE METHOD WHICH PROVIDES INPUT TO THE PROGRAM . THE FORM OF THE READ STATEMENT IS :

READ << VARIABLE,....,VARIABLE >>

FOR EVERY VARIABLE IN THE READ LIST THERE MUST BE A CORRESPONDING ELEMENT IN A DATA STATEMENT . THE READ AND DATA STATEMENTS ARE USED TOGETHER TO ASSIGN INPUT VALUES TO PROGRAM VARIABLES.

WHEN THE READ STATEMENT IS EXECUTED , EACH VARIABLE IS ASSIGNED SUCCESSIVE NUMBERS FROM A STACK OF NUMERIC DATA OR SUCCESSIVE 'STRINGS' FROM A STACK OF ALPHA-NUMERIC DATA. AS EACH VARIABLE IS READ , IT TAKES THE TOP ELEMENT OF THE APPROPRIATE DATA STACK .

go

D. DATA

THE DATA STATEMENT IS A LIST OF INPUT NUMBERS OR 'STRINGS' THAT WILL BE ASSIGNED TO VARIABLES IN A READ STATEMENT . THE FORM OF THE DATA STATEMENT IS :

DATA << NUMBER OR 'STRING',...., NUMBER OR 'STRING' >>

THE 'STRING' OF ALPHA-NUMERIC CHARACTERS MUST BE ENCLOSED IN SINGLE QUOTES .

DATA STATEMENTS MAY BE PLACED ANYWHERE IN A PROGRAM , BUT THERE IS AN UPPER LIMIT OF 500 NUMERIC AND 500 ALPHA-NUMERIC DATA ELEMENTS FOR EACH PROGRAM .

go

WHEN THE FIRST DATA STATEMENT IS INTERPRETED BY THE CAIBASIC COMPILER A FIRST IN , FIRST OUT STACK IS FORMED FOR NUMERIC AND ALPHA-NUMERIC DATA . AS EACH NUMBER OR STRING IN A DATA LIST IS INTERPRETED , IT IS PLACED ON THE BOTTOM OF ITS RESPECTIVE DATA STACK . AS OTHER DATA STATEMENTS ARE LOCATED IN THE PROGRAM ITS ELEMENTS ARE PLACED ON THE BOTTOM OF THE PROPER STACK .

1.) EXAMPLE :

```
DATA 10.0,13.33,'J.E.SMITH',18
DATA 'Z.X. DOE',19
```

PRODUCES THE FOLLOWING DATA STACKS :

NUMERIC	ALPHA-NUMERIC
10.0	J.E.SMITH
13.33	Z.X. DOE
18	
19	

go

DURING EXECUTION OF READ STATEMENTS , AS THE VARIABLES IN THE READ LIST ARE ASSIGNED VALUES FROM THE TOP OF THE APPROPIRATE DATA STACK , THE DATA STACK IS DECREMENTED AND THE NEXT ELEMENT POPS UP .

2.) EXAMPLE :

```
READ A,B1,Z$
```

ASSUMING THAT THE DATA FROM EXAMPLE 1.) IS AVAILABLE , THE VARIABLES IN THE READ LIST ARE ASSIGNED VALUES AS FOLLOWS :

```
 A <-- 10.0
B1 <-- 13.33
Z$ <-- J.E.SMITH
```

THE RESULTING DATA STACKS ARE AS FOLLOWS :

NUMERIC	ALPHA-NUMERIC
18	Z.X. DOE
19	

go

E. RESTORE , RESTORE$

THE RESTORE , RESTORE$ STATEMENTS ARE USED TO RETURN THE NUMERIC AND ALPHA-NUMERIC DATA STACKS TO THEIR ORIGINAL CONDITION SO

THAT THE DATA MAY BE USED AGAIN . THE FORM OF THE RESTORE STATEMENT
IS :

```
          RESTORE    ( RESTORES NUMERIC DATA )

          RESTORE$    ( RESTORES ALPHA-NUMERIC DATA )
```

YOU WILL NOW BE ASKED SOME QUESTIONS ABOUT THE READ , DATA AND
RESTORE STATEMENTS :

          CONSIDER THE FOLLOWING STATEMENTS AS PART OF A BASIC PROGRAM:

```
     DATA 2,5,7,16,'ANS='
     DATA 'CORRECT',25,'WRONG',0.0
     READ A,B3,I$,C1,D
     READ J$,E4,K$,X
```

go

          1.) WHAT IS THE VALUE OF C1 ?? REPLY WITH VALUE

7

          NOW CONSIDER THAT THE FOLLOWING STATEMENTS
WERE ADDED TO THE ABOVE PROGRAM :

```
     RESTORE
     READ Z4,A
```

          2.) WHAT IS THE VALUE OF Z4 ?? REPLY WITH VALUE .

16

          THE CORRECT ANSWER IS Z4=2 . THE RESTORE COMMAND
RETURNS THE NUMERIC DATA STACK TO ITS ORIGINAL CONDITION, AND
THE READ STATEMENT ASSIGNES VALUES FROM THE TOP OF THE DATA
STACK TO THE VARIABLES IN THE READ LIST AS FOLLOWS :

```
     Z4 <-- 2
     A  <-- 5
```

          F. SUMMARY
NOW THAT YOU HAVE SEEN HOW READ AND DATA STATEMENTS WORK TOGETHER
TO INPUT VALUES INTO YOUR PROGRAM , AND HOW THE PRINT STATEMENT
IS USED TO OUTPUT AND LABEL RESULTS YOU HAVE THE FACILITY
TO WRITE SIMPLE PROGRAMS USING INPUT DATA .
     FOR EXAMPLE
 ((8**2)**.5) COULD BE WRITTEN IN SYMBOLIC FORM AND THE DATA

COULD BE READ IN AS FOLLOWS:

```
READ A,B
PRINT'ANS=',(( A**B)**.5)
DATA 8,2
END
```

IN THE NEXT LESSON YOU WILL BE SHOWN HOW TO USE ASSIGNMENT STATEMENTS
THIS WILL GIVE YOU GREATER FLEXIBILITY IN WRITING EXPRESSIONS
AND WILL ALLOW YOU TO DO ASSIGNMENTS SUCH AS :

A=B**2-4*A*C , AND Z=(A**2+3)/A , THEN BY SAYING PRINT A,Z
THE RESULTS OF BOTH EXPRESSIONS WOULD BE DISPLAYED.

go

YOU WILL NOW BE GIVEN SOME REPRESENTATIVE PROBLEMS
TO GIVE YOU A CHANCE TO EXERCISE YOUR NEW PROGRAMMING TOOLS

1.) WRITE A PROGRAM TO SOLVE THE EQUATION X**2+10Y-24
WHERE THE INPUT DATA IS X=10 ,Y=3 .

2.) WRITE A PROGRAM TO SOLVE THE QUADRATIC EQUATION

(-B+(B**2-4AC)**.5)/2A
INPUT DATA IS A=2 ,B=5 ,C=2 .

IF YOU WISH TO WRITE AND EXECUTE THESE PROGRAMS NOW , REPLY : YES
AND YOU WILL ENTER THE CAIBASIC COMPILER ; OTHERWISE REPLY : NO
AND YOU WILL GO ON TO THE NEXT LESSON .

no

IF YOU DECIDE TO RUN THESE PROBLEMS LATER THE ANSWERS
WILL NOW BE GIVEN SO YOU MAY CHECK YOUR RESULTS :

1.) 106
2.) -.500

DO YOU WANT TO TERMINATE YOUR INSTRUCTION SESSION ?
(REPLY : YES OR NO)

no

IF YOU WANT TO EXECUTE PROGRAMS AT THIS TIME
THEN REPLY : YES ; OTHERWISE REPLY : NO AND THE INSTRUCTION
WILL CONTINUE .

no

```
EXECUTION BEGINS...

      *** LESSON 3. ***

THIS INSTRUCTION SET WILL INTRODUCE YOU TO ASSIGNMENT STATEMENTS
AND BUILT-IN FUNCTIONS.THE LET STATEMENT AND THE 10 BUILT IN
FUNCTION WILL ENABLE YOU TO EVALUATE AND ASSIGN VARIABLES TO
 COMPLEX ARITHMETIC EXPRESSIONS.


go

           A. LET
     THE LET STATEMENT IS AN ASSIGNMENT OR SUBSTITUTION COMMAND. IT
CAUSES THE EVALUATION OF AN EXPRESSION TO BE SUBSTITUTED FOR THE
CURRENT VALUE OF A VARIABLE.THE FORM OF THE LET STATEMENT IS:

          LET << VARIABLE >> = << EXPRESSION >>       OR

          LET << VARIABLE >> = << VARIABLE >> = .... = << EXPRESSION >>


go

      THERE MAY BE ANY NUMBER OF VARIABLE = VARIABLE
IN THE FORM OF THE LET STATEMENT . AN EXPRESSION IS A NUMBER ,
 VARIABLE , OR AN ARITHMETIC EXPRESSION .

          WHEN THE LET STATEMENT IS EXECUTED THE EXPRESSION ON THE RIGHT
SIDE OF THE EQUAL SIGN IS EVALUATED AND THE RESULTING VALUE IS
ASSIGNED TO THE VARIABLE OR VARIABLES ON THE LEFT SIDE OF THE EQUAL
SIGN . THE PREVIOUS VALUE ASSIGNED TO THE VARIABLE OR VARIABLES
WILL BE LOST . FOR EXAMPLE :

          LET A=12.3
          LET B=14.4
          LET A=B=25

THE VALUE OF THE VARIABLES A AND B IS NOW A=B= 25 .


go

           THE ONLY RESTRICTION ON THE USE OF VARIABLES
IS THAT  SIMPLE AND SUBSCRIPTED VARIABLES CAN ONLY BE ASSIGNED
NUMERIC VALUES , AND ALPHA VARIABLES CAN ONLY BE ASSIGNED ALPHA-
NUMERIC'STRINGS'. FOR EXAMPLE :

          LET A=D4=X(5)=(3**2 - 6)/4
          LET A$=X$='HELP'
```

```
          LET Y=(B - 4*A*C)**.5
          LET O$='ANSWER ='
THE FOLLOWING ASSIGNMENT IS ILLEGAL :

          LET A=D$=(3+4)

go

          YOU WILL NOW BE ASKED QUESTIONS CONCERNING
WHAT YOU HAVE JUST LEARNED :


           1.)  REPLY WITH VALUE OF X IN BELOW PROGRAM .

          LET A=4
          LET B=6
          LET C=3
          LET X=A**2 - 4*B + 3*C
          PRINT'ANSWER =',X
          END

1

           2.) REPLY WITH VALUE OF Z IN BELOW PROGRAM .

          DATA 1,2,3,4,6
          DATA'RIGHT ON',7,8
          READ A,B,X
          READ G$,Y,C,D
          LET Z=X+Y
          END

6

 YOUR RESPONSE WAS INCORRECT . THE VARIABLES ARE
ASSIGNED VALUES AS FOLLOWS :
     NUMERIC              ALPHA
      A<-- 1              Y$<-- RIGHT ON
     B<-- 2
     X<-- 3
     Y<-- 4
     C<-- 6
     D<--7

     Z <-- X+Y    ;  Z <-- 7


          3.)  LET R= A+B/C-D
WHICH FORMULA DOES THIS     STATEMENT REPRESENT ?
```

```
R
 EPLY WITH CORRECT LETTER
A.) R=(A+B)/(C-D)
B.) R= A+(B/C)-D
```

b

```
          B. BUILT-IN FUNCTIONS
     BUILT-IN FUNCTIONS ARE COMMONLY USED PROGRAMS ALREADY WRITTEN
AND STORED IN THE CAIBASIC COMPILER FOR YOUR USE . THERE ARE
FUNCTIONS TO FIND SQUARE ROOTS , LOGARITHMS , ABSOLUTE VALUES ,
AND TRIGONOMETRIC VALUES . THE FORM FOR THE BUILT-IN FUNCTIONS IS :

          FUNCTION NAME << (EXPRESSION) >>
WHERE THE EXPRESSION IS ENCLOSED IN PARENTHESIS .
```

go

```
           THE BUILT-IN FUNCTIONS AND DEFINITIONS ARE :

     SQR(X)      --- SQUARE ROOT OF ARGUMENT (MUST BE POSITIVE)
     ABS(X)       --- ABSOLUTE VALUE OF ARGUMENT
     LOG(X)       --- NATURAL LOGARITHM OF ARGUMENT
     EXP(X)      --- EXPONENTIAL FUNCTION , VALUE OF 2.718218 ** X
     INT(X)       --- INTEGER PART OF ARGUMENT IS RETURNED
     SIN(X)       --- SINE OF THE ARGUMENT
     COS(X)       --- COSINE OF ARGUMENT
     TAN(X)       --- TANGENT OF ARGUMENT
     ATN(X)       --- ARCTANGENT IN RADIANS OF ARGUMENT

IN ALL THE ABOVE BUILT-IN FUNCTIONS THE ARGUMENT IS ANY LEGAL EXPRESSION ;
AND AS NOTED THE SQR FUNCTION  REQUIRES A POSITIVE ARGUMENT ,
THE TRIGONOMETRIC FUNCTIONS REQUIRE AN ARGUMENT VALUE IN RADIANS .
```

go

```
           THE BUILT-IN FUNCTIONS ARE USED BY SIMPLY
CALLING THEM WITH THE APPROPRIATE FUNCTION NAME AND ARGUMENT .
THESE BUILT-IN FUNCTIONS ARE CONSIDERED TO BE EXPRESSIONS ,
AND THEY MAY BE USED ANY PLACE WHERE AN EXPRESSION IS LEGAL ,
 FOR EXAMPLE :

     LET Z=SQR(ABS(-5))    IS A CORRECT USE OF BUILT IN FUNCTIONS.
```

go

THE FOLLOWING PROGRAM IS A EXAMPLE OF HOW TO
USE BUILT-IN FUNCTIONS :

```
     REM  PROGRAM TO COMPUTE SQUARE ROOT AND LOGARITHMS
     READ A
     LET Y=SQR(A)
     LET Z=LOG(A)
     PRINT'SQUARE ROOT=',Y,'LOG=',Z
     DATA 4,6,8
     END
```

PRODUCES RESULT

```
     SQUARE ROOT= 2.0           LOG= 1.386
```

go

YOU WILL NOW BE ASKED QUESTIONS CONCERNING
WHAT YOU HAVE JUST LEARNED :

1.) IS THE FOLLOWING STATEMENT LEGAL
REPLY : YES , OR NO.

```
     LET Z1=4*SQR(3.0 * EXP(9.3))
```

yes

2. IS THE FOLLOWING SEQUENCE OF PROGRAM STATEMENTS LEGAL
REPLY: YES OR NO.

```
     LET B=-9
     LET X=SQR(B)
          ...
          ...
          ...
```

no

NO , THE SQUARE ROOT OF A NEGATIVE NUMBER IS AN
UNDEFINED OPERATION . IN THE NEXT LESSON YOU WILL BE SHOW A
BASIC STATEMENT FOR TESTING AND BRANCHING TO ANOTHER SEGMENT
OF THE PROGRAM IF THE TEST IS TRUE . FOR EXAMPLE THE ABOVE
PROGRAM SEQUENCE MIGHT BE ALTERED AS FOLLOWS :

```
     LET B=-9
     IF B LT 0 THEN 100
     50 LET X=SQR(X)
          ...
```

```
      :::
      :::
100 REM  NEGATIVE ARGUMENT
LET B=ABS(B)
GO TO 50
      :::
      :::
```

THIS PROGRAM SEQUENCE TESTS FOR A NEGATIVE ARGUMENT , IF TRUE
IT BRANCHES TO STATEMENT NUMBER 100 , MAKES THE ARGUMENT POSITIVE
AND BRANCHES BACK TO STATEMENT 50 TO COMPLETE THE PROGRAM .

go

C. SUMMARY

WITH THE LET STATEMENT AND BUILT-IN FUNCTIONS , PLUS THE PREVIOUS
BASIC STATEMENTS (REM , READ , DATA , PRINT) , YOU ARE
FAST GAINING AN EFFECTIVE REPERTOIRE FOR PROGRAMMING USE . IN
THE NEXT LESSON YOU WILL LEARN HOW TO SET UP LOOPS IN A PROGRAM
SO THAT THE MAIN BODY OF A PROGRAM MAY BE EXECUTED AS OFTEN
AS DESIRED . AS YOU ARE DOING YOUR REVIEW PROBLEMS , THINK ABOUT
HOW YOU COULD SET UP A LOOP TO READ IN ANY AMOUNT OF DATA ,
PROCESS IT AND THEN HALT FOR SOME TEST CONDITION .

go

THE FOLLOWING REVIEW PROBLEMS WILL EXERCISE
YOUR PROGRAMMING SKILLS TO DATE :

1.) WRITE A PROGRAM TO COMPUTE THE PRESENT WORTH OF AN INVESTMENT FOR
SOME NUMBER OF YEARS HENCE . THE FORMULA IS :

P=S(1/((1+I)**N))

WHERE P IS THE PRESENT WORTH OF A PAYMENT S IN N YEARS HENCE
AT AN INTEREST RATE OF I . FOR DATA USE I=.08 , S=5000 , N=20 .

2.) WRITE A PROGRAM TO FIND SIDES A , AND C OF A TRIANGLE USING
THE LAW OF SINES FORMULA :

A/SIN(A) = B/SIN(B) = C/SIN(C)

WHERE THE NUMERATOR IS THE SIDE AND THE DENOMINATOR IS THE
SINE OF THE ANGLE , THE CONVERSION FACTOR FROM DEGREES TO RADIANS IS:

1 DEGREE =(3.1416/180) RADIANS . THE DATA FOR THE PROGRAM IS :

SIDE B=34.91 IN. ANGLE A=98.71 DEG.
ANGLE B=49.97 DEG. ANGLE C=31.32 DEG.

IF YOU WISH TO RUN THESE PROGRAMS NOW , THEN REPLY : YES
OTHERWISE REPLY : NO.

no

```
 THE ANSWERS TO THE PROBLEMS WILL NOW BE GIVEN SO
THAT YOU MAY CHECK YOUR RESULTS LATER :

    1.) $1072.74
     2.) SIDE A=45.06 IN. AND SIDE C=23.69 IN.


          DO YOU WANT TO TERMINATE YOUR INSTRUCTION SESSION ?
 (REPLY : YES OR NO )


yes
R; T=4.84/16.96 14.03.44
```

```
EXECUTION BEGINS...

       *** LESSON 4 ***

THIS INSTRUCTION SEQUENCE WILL COVER BRANCHES . AS YOU HAVE
SEEN FROM PREVIOUS PROGRAMS , A PROGRAM'S EXECUTION USUALLY
TAKES A DIRECT ROUTE FROM THE FIRST TO THE LAST STATEMENT .
THE CONDITION THAT ALLOWS DETOURS TO OCCUR IN PROGRAMS IS CALLED
BRANCHING . THERE ARE TWO TYPES OF BRANCHING : UNCONDITIONAL
AND CONDITIONAL .


go

      A. UNCONDITIONAL BRANCHES
     AN UNCONDITIONAL BRANCH IS AN IMPERATIVE TRANSFER OF CONTROL
FROM ONE POINT IN A PROGRAM TO ANOTHER . THERE ARE TWO FORMS
OF THE UNCONDITIONAL BRANCH : THE 'GO TO' AND THE 'COMPUTED
GO TO'

          1.) GO TO << STATEMENT NUMBER >>

THIS COMMAND TRANSFERS PROGRAM CONTROL DIRECTLY TO THE STATEMENT NUMBER
AND CONTINUES EXECUTION FROM THAT POINT . THE 'GO TO'IS USED
FOR FORMING LOOPS IN A PROGRAM .


go

 A SAMPLE LOOP FOLLOWS :

     REM PROGRAM TO COMPUTE PRESENT WORTH
     REM P=INVESTMENT,S=PRINCIPAL,I=INTEREST RATE,N=NR.YEARS
     PRINT'INVESTMENT','PRINCIPAL','INTEREST','NR.YEARS'
     10 READ S,I,N
     LET P=S*(1/((1+I)**N))
     PRINTP,S,I,N
     GO TO 10
     DATA 5000,.08,20,5000,.08,10,5000,.06,10
     END
PRODUCES THE OUTPUT :

INVESTMENT          PRINCIPAL          INTEREST          NR.YEARS
  1072.74              5000               .08                20
  2315.97              5000               .08                10
  2791.99              5000               .06                10

*** ERROR , YOU TRIED TO READ MORE NUMERIC DATA THAN YOU PUT IN ***
```

```
go

 THE ERROR OCCURS BECAUSE YOU RUN OUT OF DATA DURING
THE EXECUTION OF THE LOOP SET-UP BY THE UNCONDITIONAL TRANSFER .
IF THE READ STATEMENT WERE NOT IN THE LOOP TO CAUSE THE
PROGRAM TO STOP . THEN YOU WOULD BE IN AN 'INFINITE LOOP' ,
A CONDITION IN WHICH THERE IS NO WAY TO STOP , YOU MUST ALWAYS
CHECK FOR THE 'INFINITE LOOP'CONDITION BY MAKING SURE THAT
YOUR PROGRAM HAS AN EXIT .


go

  2.) ON << EXPRESSION >> GO TO << STATEMENT NUMBER ,....., STATEMENT NUMBER >>

THIS SPECIAL FORM OF THE 'GO TO' COMMAND ,
IS CALLED THE 'COMPUTED GO TO', THE EXPRESSION
IN THE FORM OF THE STATEMENT MUST EVALUATE TO AN INTEGER NUMBER
BETWEEN 1-->9999 . IF IT IS NOT AN AN INTEGER , OR OUTSIDE THIS
RANGE AN ERROR WILL OCCUR .
     WHEN THE'COMPUTED GO TO'IS EXECUTED THE EXPRESSION IS EVALUATED ,
AND PROGRAM CONTROL TRANSFERS TO THE N-TH STATEMENT NUMBER ,
WHERE N-TH REPRESENTS THE  VALUE OF THE EXPRESSION . FOR EXAMPLE :

     LET I=3
     ON I GO TO 100,33,475,9999


go

 EXECUTION OF THE'COMPUTED GO TO' WOULD CAUSE
PROGRAM CONTROL TO TRANSFER UNCONDITIONALLY TO STATEMENT NUMBER 475 .
     YOU MUST BE CAREFUL WHEN USING THE 'COMPUTED GO TO'
NOT ONLY BECAUSE OF INFINITE LOOPS ; BUT BECAUSE THE EXPRESSION
MUST BE AN INTEGER BETWEEN 1-->9999 , AND THERE MUST BE A
STATEMENT NUMBER FOR 'ALL' POSSIBLE VALUES OF THE EXPRESSION .


go

      B. CONDITIONAL BRANCHING
     THE CONDITIONAL BRANCH TRANSFERS CONTROL ONLY IF CERTAIN
RELATIONS ARE TRUE . IF THE THE TEST OF RELATIONS IS TRUE THEN
TRANSFER OF CONTROL OCCURS ; OTHERWISE PROGRAM CONTROL CONTINUES
WITH THE NEXT STATEMENT . THE FORM OF THE CONDITIONAL BRANCH IS:

 IF << EXPRESSION >> << RELATION >> << EXPRESSION >> THEN << STATEMENT NUMBER >>

NOTE THAT ALPHA VARIABLES ARE NOT ALLOWED AS AN EXPRESSION IN A
 CONDITIONAL BRANCH .
```

```
go

      THE RELATIONS ARE :

 SYMBOLS    EXAMPLE      MEANING

  GT        A GT B       A GREATER THAN B
  GE        A GE B       A GREATER THAN OR EQUAL TO B
  LT        A LT B       A LESS THAN B
  LE        A LE B       A LESS THAN OR EQUAL TO B
  NE        A NE B       A NOT EQUAL TO B
   =        A = B        A EQUAL B


go

      WHEN THE CONDITIONAL STATEMENT IS EXECUTED THE
( EXPRESSION RELATION EXPRESSION ) IS TESTED , AND IF THE RELATION
IS TRUE , THEN CONTROL IS TRANSFERRED TO THE STATEMENT NUMBER.
OTHERWISE PROGRAM CONTROL CONTINUES TO THE NEXT SEQUENTIAL
STATEMENT , FOR EXAMPLE :

     1 READ A
     ...
     ...
     IF A LT 0 THEN 90
     LET X=SQR(A)
     ...
     ...
    90 PRINT'ILLEGAL ARGUMENT',A
     GO TO 1
    ...

THE ONLY TIME THAT THE CONDITIONAL BRANCH IS EXECUTED IS WHEN
A IS LESS THAN 0 .


go

 YOU WILL NOW BE ASKED SOME QUESTIONS ABOUT BRANCHING :
        1.) ARE THE FOLLOWING STATEMENTS CORRECT
REPLY: YES OR NO)

               GO TO END


no

                IF X1 LT A$ THEN 10


no
```

```
            IF(X**3-4) GE 10 THEN GO TO 100
```

yes

```
 YOUR ANSWER IS INCORRECT . THE ONLY THING ALLOWED
AFTER THEN IS A STATEMENT NUMBER .
```

```
            LET Y=6,
            ON Y GO TO 1,3,5,7,999
```

no

```
          2.) CONSIDER THIS PROGRAM SEGMENT , ANY ERRORS (REPLY: YES OR NO).

     1 READ A,B
     3 LET Z=A*B
     IF Z LT 0 GO TO  999
     ...
     ...
     GO TO 3
     DATA 3,5,7,10
     999 END
```

yes

```
 THERE IS AN 'INFINITE' LOOP IN THIS PROGRAM . IT
COULD BE CORRECTED BY CHANGING THE GO TO 3 , TO GO TO 1 .
      WHEN YOU WRITE PROGRAMS HAVING BRANCHES OR LOOPS YOU MUST ALWAYS
CONSIDER HOW THE PROGRAM WILL STOP. YOU HAVE OBSERVED THAT
AN 'INFINITE LOOP'CAN BE STOPPED BY HAVING A READ STATEMENT
IN THE LOOP AND JUST RUN OUT OF DATA . HOWEVER, ENDING A PROGRAM
ON AN ERROR IS AN INELIGANT METHOD . THE MOST COMMON METHODS
USE THE 'GO TO' AND ' IF/THEN' COMMANDS TO CONTROL PROGRAM
LOOPS AND ARE AS FOLLOWS :
```

go

```
      1. COUNT AND TEST METHOD , IN WHICH A COUNTER
IS INCREMENTED IN THE LOOP , AND WHEN THE COUNTER REACHES A
CERTAIN VALUE , BRANCH OUT OF THE LOOP . EXAMPLE PROGRAM

          REM COUNT AND TEST METHOD
          REM N IS COUNTER , INITALIZED TO 0 , AND COUNTS FROM 1-->10
```

```
          LET N=Z=0
          READ X
          10 LET Z=Z+X
          LET N=N+1
          IF N GT 10 THEN 100
          GO TO 10
          100 PRINT'SUM=',Z
          DATA 10
          END
```

REPLY WITH VALUE OF Z .

11

YOUR ANSWER IS INCORRECT . THE ONLY WAY TO BE SURE OF VALUES IN A LOOP IS TO SET UP A TABLE OF THE VARIABLES AND KEEP TRACK OF THERE VALUES IN THE LOOP :

X	N	Z
10	0	0
	1	10
	2	20
	3	30
	...	
	...	
	9	90
	10	100
	11	110

THUS Z=110

2. READ AND TEST METHOD , IN WHICH A VALUE IS READ IN AND TESTED FOR THE END OF LOOP CONDITION .

```
          REM READ AND TEST DEMONSTRATION
          REM 9999 IS END VALUE
          LET Z=0
          1 READ X
          IF X=9999 THEN 9999
          LET Z=Z+X
          GO TO 1
          DATA 1,2,3,4,5,6,9999
          9999 PRINT 'SUM=',Z
          END
```

REPLY WITH VALUE OF Z

21

C. SUMMARY

YOU HAVE SEEN HOW THE UNCONDITIONAL BRANCHES ('GO TO' AND 'COMPUTED GO TO') TRANSFER PROGRAM CONTROL , AND HOW THE CONDITIONAL BRANCH ('IF/THEN') TESTS FOR TRANSFER OF PROGRAM CONTROL ; AND YOU HAVE OBSERVED THE CONTROL OF LOOPS SO THAT A PROGRAM SEGMENT MAY BE REPEATED UNTIL A SPECIFIED CONDITION IS MET. IN THE NEXT LESSON YOU WILL LEARN A BASIC STATEMENT TO CONTROL A LOOP BY THE INCREMENT AND TEST METHOD .

THE FOLLOWING PROBLEMS WILL TEST YOUR NEW SKILLS :

go

1.) WRITE A PROGRAM TO COUNT THE NUMBERS BETWEEN 50 AND 60 , AND ALSO PRINT THEM OUT . THE INPUT DATA IS 10,50,35,75,62,60,54,54 .

2.) WRITE A PROGRAM TO COMPUTE THE PRESENT WORTH OF AN INVESTMENT FOR SOME YEARS HENCE , AT VARING INTEREST RATES . THE FORMULA IS:

P=S(1/((1+I)**N))

WHERE P=PRESENT WORTH,S=PRINCIPAL,I=INTEREST,AND N=NR.OF YEARS.
FOR DATA USE S=5000,AND I=.04-->.8, IN INCREMENTS OF .01 , AND N=20

IF YOU DESIRE TO EXECUTE THESE PROGRAMS NOW
REPLY: YES ; OTHERWISE REPLY : NO

no

DO YOU WANT TO TERMINATE YOUR INSTRUCTION SESSION ?
(REPLY : YES OR NO)

no

IF YOU WANT TO EXECUTE PROGRAMS AT THIS TIME
THEN REPLY : YES ; OTHERWISE REPLY : NO AND THE INSTRUCTION WILL CONTINUE .

no

REPLY WITH THE NUMBER OF THE LESSON YOU WISH TO COVER

```
EXECUTION BEGINS...

      *** LESSON 5 ***
THIS LESSON WILL INTRODUCE YOU TO ITERATION , SUBSCRIPTED
VARIABLES , AND  LISTS (VECTORS) AND TABLES (MATRICES).

     A. ITERATION (LOOPING)
     IN THE LAST LESSON YOU WERE SHOWN HOW TO USE CONDITIONAL AND
UNCONDITIONAL BRANCHES TO CONTROL LOOPING . THE COUNT AND TEST
ITERATIVE LOOP OCCURS SO FREQUENTLY THAT AN ABBREVIATED BASIC
STATEMENT HAS BEEN DEVISED TO CONTROL LOOPING . THE ITERATIVE
LOOP HAS THE FOLLOWING FORM :

FOR << SIMPLE VARIABLE >> = << EXPRESSION >> TO << EXPRESSION >>
                    STEP << EXPRESSION >>
            ...
           ...
           ...
          NEXT << SIMPLE VARIABLE >>


go

 FOR EXAMPLE CONSIDER THIS PROGRAM SEGMENT :

     FOR I=1 TO 10 STEP 2
     LET X=X+I
     NEXT I

     THE SIMPLE VARIABLE FOLLOWING 'FOR'IS THE LOOP INDEX . WHEN
THE FOR/NEXT PAIR IS EXECUTED , THE LOOP INDEX IS GIVEN THE
VALUE (INITIALIZED) OF THE FIRST EXPRESSION (I=1 IN EXAMPLE).
THIS INDEX IS THEN TESTED TO DETERMINE WHETHER IT IS'GREATER THAN'
THE SECOND EXPRESSION AFTER 'TO'(10 IN EXAMPLE).IF IT IS
GREATER, CONTROL IS TRANSFERRED TO THE STATEMENT FOLLOWING
'NEXT'.OTHERWISE THE REMAINING STATEMENTS WITHIN THE LOOP
(FOR/NEXT) ARE EXECUTED SEQUENTIALLY UNTIL THE 'NEXT' STATEMENT
IS REACHED.


go

 WEN THE 'NEXT' STATEMENT IS REACHED , THE LOOP
INDEX IS INCREASED(INCREMENTED) BY THE AMOUNT OF THE EXPRESSION
FOLLOWING'STEP', AND CONTROL IS TRANSFERRED BACK TO THE'FOR'
STATEMENT WHERE THE LOOP CONTINUES UNTIL THE INDEX VALUE IS
```

GREATER THAN THE FINAL VALUE . FOR EXAMPLE :

```
REM DEMO LOOP. SUM THE NUMBERS FROM 1-->10.
LET C=0
FOR I=1 TO 10 STEP 1
LET C=C+1
NEXT I
PRINT'SUM=',C
END
```

go

YOU WILL NOTICE THAT THE SIMPLE VARIABLE FOLLOWING 'NEXT' IS THE SAME AS THE SIMPLE VARIABLE FOLLOWING'FOR', AND THAT THE'NEXT' STATEMENT MARKS THE END OF THE LOOP .

BECAUSE THE INCREMENT VALUE OF A LOOP IS COMMONLY ONE(1) , THE 'STEP' MODIFIER AND ITS EXPRESSION MAY BE OMITTED , AND THE INCREMENT VALUE WILL BE ASSUMED TO BE ONE(+1). FOR EXAMPLE THE ABOVE 'FOR' STATEMENT COULD BE WRITTEN :

```
FOR I=1 TO 10
```

THE INCREMENT VALUE AFTER'STEP' MAY BE POSITIVE OR NEGATIVE ALLOWING THE FLEXIBILITY OF LOOPING FORWARD OR BACKWARD . FOR A NEGATIVE'STEP'VALUE THE TEST BECOMES 'LESS THAN'. FOR EXAMPLE THE FOLLOWING'FOR'STATEMENTS ARE EQUIVALENT :

```
FOR I=1 TO 10
FOR I=10 TO 1 STEP -1
```

go

ANOTHER USEFUL TECHNIQUE OF LOOPING IS 'NESTING' NESTING REFERS TO PLACING ONE LOOP INSIDE ANOTHER LOOP . THE INNER LOOP 'SPINS'AROUND AS MANY TIMES AS THE OUTER LOOP IS INCREMENTED . FOR EXAMPLE CONSIDER THIS PROGRAM SEGMENT

```
FOR I=1 TO 10
FOR J=1 TO 20 STEP 2
 ...
 ...
NEXT J
NEXT I
```

IN THIS EXAMPLE THE OUTSIDE LOOP(I) IS

REPEATED 10 TIMES,AND THE INNER LOOP(J) WOULD BE REPEATED 20
TIMES FOR EACH INCREMENT OF THE OUTSIDE LOOP, OR 200 REPETITIONS
LOOPS MAY BE NESTED UP TO A MAXIMUM OF 20 ; HOWEVER , THEY
CANNOT OVERLAP . THE INNERMOST LOOP MUST BE CLOSED WITH ITS
'NEXT' STATEMENT BEFORE ENCOUNTERING THE NEXT OUTER LOOP'S
'NEXT' STATEMENT . FOR EXAMPLE :

```
FOR X=10 TO 1 STEP -1
FOR Y=3 TO 5
FOR Z=-3 TO -5 STEP -1
 ...
 ...
NEXT Z
NEXT Y
NEXT X
```

go

WITHIN A FOR/NEXT LOOP CONDITIONAL AND UNCONDITIONAL
BRANCHES MAY BE USED TO TRANSFER CONTROL OUT OF A LOOP
OR WITHIN LIMITS OF THE SAME LOOP . HOWEVER , IT IS NOT POSSIBLE
TO BRANCH INTO THE MIDDLE OF A FOR/NEXT LOOP BECAUSE LOGIC
PROBLEMS OCCUR AND AN ERROR WILL RESULT . AN ADDITIONAL ITEM
TO BE CAREFUL ABOUT IS USING THE INDEX VARIABLE OF THE FOR/NEXT
LOOP IN COMPUTATIONS . IF YOU ALTER THE VALUE OF THE LOOP INDEX
YOU WILL EFFECT THE ACTION OF THE LOOP . FOR EXAMPLE :

```
FOR I=1 TO 10
LET I=I+10
NEXT I
PRINT I
END
```

REPLY WITH VALUE OF I THAT IS PRINTED

11

YOU WILL NOW BE ASKED SOME QUESTIONS ABOUT WHAT YOU HAVE JUST LEARNED .

1.) SAMPLE PROGRAM

```
LET S=0
FOR K=-5 TO 5
LET S=S+K
NEXT K
PRINT'SUM=',S
END
```

REPLY WITH VALUE OF S .

0

2.) SAMPLE PROGRAM

```
FOR I=100 TO 1 STEP 2
LET Z=I**2
LET Y= Z+10
```

```
               NEXT I
               PRINT I
               END
```

REPLY WITH VALUE OF I

1

YOUR ANSWER IS INCORRECT . THE INDEX VALUE OF THE
LOOP(I=100) IS GREATER THAN THE FINAL VALUE(1) IN THE FIRST
TEST . THUS I=100 . IF THE LOOP WERE DECREMENTED IN STEPS OF -2
THEN THE LOOP WOULD BE EXECUTED AND THEN I=0 .

3.) SAMPLE PROGRAM

```
    LET T=0
    FOR I=5  TO 1 STEP -1
    FOR J=1 TO 5
    IF I=J THEN 10
    GO TO 15
    10 LET T=T+1
    PRINT 1
    15 NEXT J
    NEXT I
    PRINT T
    END
```

REPLY WITH VALUE OF T .

5

B. SUBSCRIPTED VARIABLES/LISTS AND TABLES

1. IN LESSON 1 YOU LEARNED THAT SIMPLE VARIABLES WERE A LETTER OR A LETTER FOLLOWED BY A DIGIT . SUBSCRIPTED VARIABLES CONSIST OF A LETTER FOLLOWED BY A SINGLE OR DOUBLE SUPSCRIPT IN PARENTHESIS THE SUBSCRIPTS MAY BE ANY LEGAL EXPRESSION THAT EVALUATES TO AN INTEGER VALUE . FOR EXAMPLE :

A(1) , B(3) , D(1,3) , Z(I,J) , X(3**A) , D(A(1)) ARE LEGAL SUBSCRIPTED VARI

BUT A1(1) , Z(1,2,3) ARE ILLEGAL SUBSCRIPTED VARIABLES.

A USE OF SUBSCRIPTED VARIABLES FOLLOWS :

go

2. A NUMERIC LIST , OR VECTOR , OR SINGLE DIMENSIONED ARRAY , IS A SET OF NUMERIC VALUES ARRANGED IN AN ORDERLY MANNER . FOR EXAMPLE , SUPPOSE YOU WOULD LIKE TO READ SOME NUMBERS INTO YOUR PROGRAM AND HAVE THESE NUMBERS AVAILABLE AND IDENTIFIED FOR USE AT A LATER TIME . YOU COULD ASSIGN SIMPLE VARIABLES TO EACH VALUE , BUT WHAT IF YOU HAD 100 NUMBERS ? IT IS EASIER TO THINK OF THE NUMBERS AS A LIST OF VALUES OR A VECTOR THE SIZE OF WHICH IS DETERMINED BY HOW MANY NUMBERS YOU HAVE . YOU THEN SIMPLY ASSIGN THE VALUES TO THE LIST .

go

THE FOLLOWING PROGRAM WILL ASSIGN 10 VALUES TO THE LIST'A' WHICH CONTAINS 10 ELEMENTS :

```
DIM A(10)
FOR L=1 TO 10
READ A(L)
NEXT L
DATA 1,3,7,5,6,9,4,2,14,4
END
```

THE LIST A , CONTAINING 10 ELEMENTS , IS REPRESENTED BY THE SUBSCRIPTED VARIABLE A(L) . READ A(L) IS IN A FOR/NEXT LOOP , AND WITHIN THE LOOP THE LIST ELEMENTS A(1)-->A(10) ARE ASSIGNED VALUES . NOW THE LIST A HOLDS THE 10 VALUES AND EACH VALUE IS IDENTIFIABLE . CONSIDER THE FOLLOWING PROBLEM WHICH SEARCHES A LIST TO FIND THE LARGES

go

```
 DM N(10)
FOR I=1 TO 10
LET N(I)=0
NEXT I
READ K
FOR J=1 TO K
READ N(J)
NEXT J
LET L=N(1)
FOR J=2 TO K
IF N(J) LT L THEN 10
LET L=N(J)
```

```
10 NEXT J
PRINT 'LARGEST NUMBER=',L
DATA 5,3,5,9,10,6
END
```

REPLY WITH VALUE OF L

10

THE FIRST FOR/NEXT LOOP ZEROS OUT THE LIST .
IT IS A GOOD PRACTICE TO PUT SOME VALUES IN THE LISTS YOU USE
OTHERWISE THE COMPUTER MAY ASSIGN RANDOM VALUES . BY PUTTING
ZERO IN EACH ELEMENT OF THE LIST YOU ARE ASSURED THAT THE LIST
IS CLEANED UP BEFORE YOU USE IT.

go

3. IN ADDITION TO LISTS(VECTORS), BASIC ALLOWS
YOU THE ABILITY TO USE TABLES , OR MATRICES , OR TWO DIMENSIONAL
ARRAYS . THE SUBSCRIPTS OF A TABLE REPRESENT THE ROWS AND
COLUMNS OF THE TABLE. THE FIRST SUBSCRIPT IS THE ROW AND THE
SECOND SUBSCRIPT IS THE COLUMN . FOR EXAMPLE IN TABLE'D':
D(3,4) REFERS TO THE VALUE OF THE ELEMENT IN ROW 3 , COLUMN 4
OF TABLE'D'.

go

AS WITH A LIT , ANY ELEMENT OF A TABLE MAY BE
REFERENCED BY DEFINING THE PAIR OF SUBSCRIPTS AS DESIRED IN
ROW-COLUMN ORDER . IN A 3X3 TABLE(MATRIX) THE TABLE IS REFERENCED
AS FOLLOWS :

```
1,1     1,2     1,3
2,1     2,2     2,3
3,1     3,2     3,3
```

WHERE THE FIRST SUBSCRIPT IDENTIFIES THE ROW AND THE SECOND THE COLUMN .
USING THIS REFERENCE SYSTEM CONSIDER HOW TO FILL THE FOLLOWING TABLE

```
1       2       3
4       5       6
7       8       9
```

go

THS TABLE'X' COULD BE FILLED AS FOLLOWS :

```
DIMX(3,3)
FOR I=1 TO  3
FOR J=1 TO 3
READ X(I,J)
NEXT J
NEXT I
DATA 1,2,3,4,5,6,7,8,9
```

```
     END

REPLY WITH VALUE OF X(2,3)


6

 NO WITH TABLE'X' ASSIGNED VALUES CONSIDER THIS
PROGRAM (ASSUMING TABLE'X' HAS BEEN FILLED BY THE ABOVE PROGRAM)

     LET S=0
     FOR I=1 TO 3
     FOR J=1 TO 3
     IF I NE J THEN 5
     LET S=S+X(I,J)
     5 NEXT J
     NEXT I
     PRINT'SUM=',S
     END

REPLY WITH VALUE OF S


15

      YOU HAVESEEN HOW SUBSCRIPTED VARIABLES ARE
USED TO SET UP LISTS(VECTORS) AND TABLES(MATRICES). HOWEVER TO
USE LISTS AND TABLES IN A PROGRAM , YOU MUST DIMENSION THE
MAXIMUM SIZE OF YOUR LIST OR TABLE SO THAT ENOUGH SPACE WILL
BE ALLOCATED IN THE COMPUTER MEMORY. THIS DIMENSIONING IS DONE
WITH THE DIM STATEMENT.


go

           C. DIM STATEMENT

     THE DIM STATEMENT TELLS THE COMPUTER THE MAXIMUM SIZE OF VECTORS
AND TABLES THAT WILL BE USED IN YOUR PROGRAM . THE DIM STATEMENT
MUST APPEAR IN THE PROGRAM BEFORE'ANY' REFERENCE IS MADE
TO THE LIST OR TABLE . IN GENERAL PRACTICE THE DIM STATEMENT
IS USUALLY THE FIRST STATEMENT IN THE PROGRAM . THE FORM OF THE
DIM STATEMENT IS :

          DIM <<LIST VARIABLE>> ( <<SIZE>> )

          DIM <<TABLE VARIABLE>> ( <<SIZE,SIZE>> )


go
```

THE LIST AND TABLE VARIABLES ARE SUBSCRIPTED VARIABLES
AS DEFINED EARLIER . THE SIZE IS AN UNSIGNED INTEGER IN
PARENTHESIS WHICH DENOTES THE MAXIMUM SIZE OF THE LIST OR TABLE
THE DIM STATEMENT MAY CONTAIN A NUMBER OF LISTS OR TABLES
WITH THEIR SIZES SEPARATED BY COMMAS , FOR EXAMPLE :

```
DIM A(6),X(10,3) ,Z(14,21)
```

go

IF YOU TRY TO REFERENCE AN ELEMENT IN A LIST
OR TABLE BEYOND THE MAXIMUM SIZE IN THE DIM STATEMENT YOU WILL
GET AN ERROR . THE MAXIMUM SIZE LIST(VECTOR) OR TABLE(MATRIX)
ALLOWED BY THE CAI-BASIC COMPILER IN ANY ONE PROGRAM IS A TOTAL
OF 1600 COMPUTER MEMORY SPACES .
YOU WILL NOW BE ASKED SOME QUESTIONS :
1.) ARE THE FOLLOWING DIM STATEMENTS CORRECT,REPLY : YES OR NO

```
DIM A1(10)
```

yes

THE STATEMENT IS INCORRECT . SUBSCRIPTED VARIABLES
ARE A'SINGLE' LETTER FOLLOWED BY ONE OR TWO SUBSCRIPTS IN
PARENTHESIS.

```
DIM  X(500),B(1,10),C(5,5,5)
```

no

D. SUMMARY

IN THIS LESSON YOU HAVE LEARNED HOW TO USE THE FOR/NEXT STATEMENT
AND HOW TO MANIPULATE LISTS(VECTORS) AND TABLES(MATRICES)
BY USING SUBSCRIPTS VARIABLES AND THE DIM STATEMENT . YOU NOW
HAVE ALL THE TOOLS TO BEGIN WRITING SOPHISTICATED PROGRAMS ;
AND AS YOU WRITE MORE COMPLICATED PROGRAMS , YOU WILL FIND A
NEED FOR SUBROUTINES . SUBROUTINES ARE COMMONLY USED PROGRAM
SEGMENTS THAT ARE USED OVER AGAIN IN OTHER PARTS OF YOUR PROGRAM
SUBROUTINES ALLOW YOU TO BRANCH TO THE COMMONLY USED SEGMENT
AND THEN RETURN TO WHERE YOU WERE AND CONTINUE EXECUTING . THE
SUBROUTINE CAN BE'CALLED' FROM ANYWHERE IN YOUR PROGRAM . THE
GOSUB STATEMENT ALLOWS SUBROUTINES IN BASIC , AND YOU WILL BE

```
INTRODUCED TO IT IN THE NEXT LESSON .

      YOU WILL NOW BE GIVEN TWO OPTIONAL PROGRAMMING
PROBLEMS TO EXERCISE YOUR NEW TOOLS .

     1.) WRITE A PROGRAM TO REVERSE THE NUMBERS IN A  10-ELEMENT LIST'X'
IN OTHER WORDS INTERCHANGE X(1) WITH X(10),X(2) WITH X(9), ETC.
READ IN TEN VALUES AND TEST YOUR PROGRAM BY PRINTING THE LIST
BEFORE AND AFTER .

     2.) WRITE A PROGRAM TO ARRANGE THE FOLLOWING LIST IN
DECENDING ORDER : 10,30,5,15,40 . ONE METHOD OF APPROACHING THIS
IS TO CHECK THE FIRST ELEMENT OF THE LIST AGAINST THE SECOND .
IF THE FIRST IS NOT LARGER THEN EXCHANGE THE TWO , OTHERWISE
GO AND COMPARE THE NEXT TWO IN THE LIST. THIS PROCESS IS REPEATED
UNTIL THERE ARE NO MORE EXCHANGES TO BE MADE . A COUNT OF
THE EXCHANGES CAN BE MADE , AND WHEN THE COUNT EQUALS 0 THE LIST IS IN ORDER.


go

 IF YOU WANT TO EXECUTE THESE PROBLEMS REPLY : YES ,OTHERWISE  NO ,


no

      IF YOU WOULD LIKE TO SEE A SOLUTION TO THE PROBLEMS REPLY : YES , OTHERWISE NO

no

           DO YOU WANT TO TERMINATE YOUR INSTRUCTION SESSION ?
 (REPLY : YES OR NO )


no

           IF YOU WANT TO EXECUTE PROGRAMS AT THIS TIME
THEN REPLY : YES ; OTHERWISE REPLY : NO AND THE INSTRUCTION
WILL CONTINUE .


no

      REPLY WITH THE NUMBER OF THE LESSON YOU WISH TO COVER
```

```
EXECUTION BEGINS...

      *** LESSON 6 ***
THIS LESSON WILL INTRODUCE YOU TO SUBROUTINES AND THE PROGRAMMING
TECHINQUE CALLED'RECURSION'.

     A. SUBROUTINES
     IN MANY PROGRAMS A BLOCK OF STATEMENTS MAY BE NEEDED ON MORE
THAN ONE OCCASION . SINCE THE REPETITION OF A BLOCK OF STATEMENTS
IS BURDENSOM , A TECHNIQUE TO ECONOMIZE ON CODING INSTRUCTIONS
CALLED A SUBROUTINE , IS PRESENTED. THE FORM OF THE SUBROUTINE IS :

     GOSUB << STATEMENT NUMBER >>
     ...
     ...
     ...
     RETURN


go

      EXECUTION F THE GOSUB STATEMENT CAUSES THE
COMPUTER TO TRANSFER CONTROL TO THE STATEMENT NUMBER AFTER'GOSUB'
WHEN CONTROL IS TRANSFERRED , STATEMENTS ARE EXECUTED SEQUENTIALLY
UNTIL A 'RETURN' IS ENCOUNTERED . AT THAT TIME , CONTROL
IS RETURNED TO THE NEXT STATEMENT FOLLOWING THE 'GOSUB'
STATEMENT WHICH 'CALLED' THE SUBROUTINE . FOR EXAMPLE CONSIDER
THE PROGRAM SEGMENT :


go

     ...
     PRINT A,B,C,D,E
     GOSUB 500
     FOR I=1 TO E
     ...
     ...
     ...
     500 REM SUBROUTINE
     LET X=A+B
     LET Z=B+C
     LET R=SQR(D)
     RETURN
     ...


go
```

THE SUBROUTINE CONSISTS OF A SEQUENCE OF BASIC STATEMENTS ENDING WITH A 'RETURN' STATEMENT . THE SUBROUTINE MAY ONLY BE ENTERED FROM A 'GOSUB' STATEMENT , AND WILL ONLY RETURN TO ITS PROPER PLACE AFTER ENCOUNTERING A 'RETURN' STATEMENT .

ALL THE VARIABLES OF THE MAIN PROGRAM ARE AVAIABLE (PASSED) TO THE SUBROUTINE , AND VICE VERSA . THE VARIABLES IN THE SUBROUTINE MUST BE CHOSEN CAREFULLY SO AS NOT TO ACCIDENTLY CONFLICT OR ALTER VARIABLES IN THE MAIN PROGRAM .

go

THE FOLLOWING PROGRAM TO COMPUTE THE GREATEST COMMON DENOMINATOR(GCD) OF THREE NUMBERS BY EUCLID'S ALGORITHM WILL DEMONSTRATE A USE OF SUBROUTINES :

```
     REM FIND GCD OF A,B,C
     PRINT 'A','B','C','GCD'
  20 READ A,B,C
     IF C=9999 THEN 9999
     LET X=A
     LET Y=B
     REM G=GCD OF A,B
     GOSUB 200
     LET X=G
     LET Y=C
     REM G=GCD OF G,C
     GOSUB 200
     PRINT A,B,C,G
     GO TO 20
          200 LET Q=INT(X/Y)
              LET R=X-Q*Y
              IF R=0 THEN 300
              LET X=Y
              LET Y=R
              GO TO 200
          300 LET G=Y
          RETURN
     DATA 60,90,120
     DATA 38456,64872,98765
     DATA 0,0,9999
9999 END
```

OUTPUT PRODUCED :

```
    A           B           C          GCD
    60          90         120          30
 38456       64872       98765           1
```

go

```
       GOSUB STATEMENTS MAY BE NESTED TO LOGICALLY
  CALL ANOTHER GOSUB , SIMILAR TO FOR/NEXT NESTED LOOPS . YOU
MUST ONCE MORE BE CAREFUL ABOUT THE STATUS OF VARIABLES BECAUSE
VARIABLES WILL BE PASSED FROM ONE SUBROUTINE TO ANOTHER .
     A SUBROUTINE THAT IS NESTED SO THAT IT CALLS ITSELF IS CALLED
RECURSION OF SUBROUTINES OR RECURSION .


go

      8. RECURSION
     RECURSION IS A PROGRAMMING TECHNIQUE IN WHICH SUBROUTINES CAN
CALL THEMSELVES .FOR EXAMPLE CONSIDER THIS PROGRAM WHICH FINDS
HE FACTORIAL OF A NUMBER USING ITERATIVE METHODS AND THEN
USING RECURSION :
     REM FACTORIAL :  F(N)=1*2*3*....*(N-1)*N
     PRINT'ITERATIVE SOLUTION'
     PRINT'N','F(N)'
  10 READ N
     IF N GT 0 THEN 20
     GO TO 50
  20 PRINT N,
     GOSUB 100
     PRINT F
     GO TO 10
       100 REM ITERATIVE  SOLUTION
             LET F=1
             FOR I=1 TO N
             LET F=F*I
             NEXT I
          REM  RETURN F=FACTORIAL(N)
       RETURN
  50 REM FACTORIAL :  F(N)= IF N=0 THEN F(N)=1
     REM  OTHERWISE , F(N)=N*F(N-1)
     RESTORE
     PRINT'RESURSIVE SOLUTION'
  60 READ N
     IF N GT 0 THEN 70
     GO TO 9999
  70 PRINT N,
     GO SUB 200
     PRINT F
     GO TO 60
       200 REM RECURSIVE SOLUTION
             IF N GT 0 THEN 210
             LET F=1
             RETURN
       210 LET N=N-1
             REM RECURSIVE CALL
```

```
            GOSUB 200
            LET N=N+1
            LET F=F*N
      RETURN
    DATA 2,5,8,6,-1
999 END
```

go

```
 IN THE ITERATIVE METHOD THE FACTORIAL FUNCTION ,
F(N) , WAS MULTIPLIED BY ITSELF IN A LOOP FROM I=1-->N . IN THE
RECURSIVE SOLUTION  THE FACTORIAL FUNCTION ,F(N), IS DEFINED
IN TERMS OF ITS FINAL VALUE . WHEN N=0 , F(N)=1 ; OTHERWISE
F(N)=N*(FACTORIAL OF N-1) .
     TO KEEP TRACK OF WHERE THE SUBROUTINE CALLS RETURN IN RECURSION
IT HELPS TO EVISIONING A LAST-IN-FIRST-OUT (LIFO) STACK CONTAINING
THE ADDRESS OF THE GOSUB STATEMENTS. EVERY TIME A GOSUB
IS ENCOUNTERED , PUT ITS ADDRESS ON TOP OF THE STACK (PUSHING
DOWN ANYTHING PREVIOUSLY ON THE STACK). THEN EVERY TIME A RETURN
IS ENCOUNTERED , RETURN TO THE TOP ADDRESS ON THE STACK (AND
POP UP THE NEXT ADDRESS ON THE STACK).
```

go

```
 TO FULLY UNDERSTAND THE CONCEPT OF RECURSION YOU
SHOULD STEP THROUGH THE FACTORIAL PROBLEM BY HAND USING THE
HELP OF THE STACK TO SEE HOW RECURSION WORKS . IF YOU UNDERSTAND
THE CONCEPT OF RECURSION YOU ARE READY TO SOLVE THIS PROBLEM :
```

```
     LET X=1
     GOSUB 100
     PRINT X
     GO TO 9999
       100 LET X=X+1
             IF X GT 3 THEN 150
             GOSUB 100
       150 LET X=X+1
       RETURN
9999 END
```

```
REPLY WITH VALUE OF X
```

5

```
 YOUR ANWER IS INCORRECT . CONSIOER THE FOLLOWING
TABLE OF VALUES FOR X , ANO ITEMS IN THE STACK(1-FIRST GOSUB, 2-SECOND GOSUB)

     X=1            STACK
                     1
     X=2            STACK
                     2
                     1
     X=3            STACK
                     2
                     2
                     1
     X=4
     X=5            STACK
                     2
                     1
     X=6            STACK
                     1
    X=7

YOU SHOULO STEP THROUGH THIS
EXAMPLE UNTIL YOU UNDERSTANO THE RECURSION TECHNIQUE . HOWEVER,
KNOWING HOW TO USE RECURSION IS NOT A REQUIREMENT FOR KNOWING
 HOW TO USE BASIC , IT IS ONLY A CLASSIC PROGRAMMING TECHNIQUE .


      C. SMMARY
     SUBROUTINES ANO RECURSION ARE A VALUABLE ADDITION TO YOUR REPERTOIR
OF BASIC STATEMENTS . THEY SAVE BOTH PROGRAMMER TIME ANO
COMPUTER STORAGE SPACE . THE GOSUB/RETURN COMMAND ANO THE OTHER
TWELVE BASIC STATEMENTS THAT YOU HAVE ALREADY LEARNEO ANO USED
FORM THE BASIC LANGUAGE .
THE FACILITY THAT YOU GAIN IN PROGRAMMING BY USING THE BASIC
LANGUAGE WILL OEPENO UPON HOW OFTEN YOU EXERCISE YOUR SKILLS.
     THE LAST LESSON WILL PROVIDE YOU WITH A BRIEF SUMMARY OF THE
BASIC LANGUAGE .

go

      THE FOLLOWING PROBLEMS WILL TEST YOUR LATEST SKILLS :

     1. COMPUTE THE NET SALARY , RETIREMENT CONTRIBUTION
AND TAX FOR N EMPLOYEES. NET PAY=GROSS SALARY-RETIREMENT-TAX
USE SUBROUTINES TO CALCULATE :
          A) RETIREMENT CONTRIBUTION
               IF SALARY <$800  , R=$0.0
               IF SALARY <$2500  , R=$10.
               IF SALARY >$2500  , R=$20,
          B) TAX
               IF SALARY <$600  , T=$5  ; OTHERWISE
               T=.03(SALARY)
THE DATA IS:
    EMPLOYEE          GROSS SALARY
```

```
JONES                3000
SMITH                5500
BROWN                475
DALEY                10,500
BERRY                4300
```

2. WRITE A RECURSIVE PROGRAM TO SUM THE NUMBERS FROM X TO Y . READ IN YOUR OWN DATA AND TEST YOUR PROGRAM .

go

IF YOU WANT TO EXECUTE THESE PROGRAMS NOW , THEN REPLY : YES ,OTHERWISE NO

no

DO YOU WANT TO TERMINATE YOUR INSTRUCTION SESSION ? (REPLY : YES OR NO)

no

IF YOU WANT TO EXECUTE PROGRAMS AT THIS TIME THEN REPLY : YES ; OTHERWISE REPLY : NO AND THE INSTRUCTION WILL CONTINUE .

no

REPLY WITH THE NUMBER OF THE LESSON YOU WISH TO COVER

```
EXECUTION BEGINS...

      *** LESSON 7 ***

THIS LESSON SUMMARIZES THE BASIC DEFINITIONS AND BASIC STATEMENTS
THAT YOU HAVE LEARNED IN CAI-BASIC .

     A. BASIC DEFINITIONS

1. ALPHA-NUMERIC CHARACTERS :

     A) DIGITS : 0-->9

     B) LETTERS : A-->Z

     C) SPECIAL CHARACTERS :  ** * / + - ( ) = ' , $

2. STATEMENT NUMBERS : ONE TO FOUR DIGITS (0-->9999)

3. STRING : ANY SEQUENCE OF ALPHA-NUMERIC CHARACTERS ENCLOSED
IN 'SINGLE' QUOTES .  EG. 'HELP'


go

 4. NUMBERS : (LIMITED TO 9 DIGITS)

     A) INTEGERS :  DIGITS WITH NO FRACTIONAL PART . EG. 5,7,10

     B) REAL : DIGIT WITH A FRACTIONAL PART . EG. 5.0,7.31,-16.0
A NUMBER MAY BE PRECEEDED BY A SIGN (+,-) , BUT IS ASSUMED TO
BE POSITIVE IF NONE IS GIVEN .

5. VARIABLES :

     A) SIMPLE VARIABLES : A SINGLE LETTER , OR A SINGLE LETTER
FOLLOWED BY A DIGIT . EG  A,A1,B6,Z0

     B) ALPHA VARIABLES : A SINGLE LETTER FOLLOWED BY A DOLLAR SIGN ($)
     EG. A$,V$

     C) SUBSCRIPTED VARIABLE : (SINGLE LETTER)

          (1) SINGLE SUBSCRIPT :  <<LETTER>> ( <<EXPRESSION>> )

          (2) DOUBLE SUBSCRIPT :  <<LETTER>> ( <<EXPRESSION,EXPRESSION>> )

EG
```

```
    A(5),Z(5,10),X(A*B,X**2)

go

 5. OPERATORS : (LISTED IN DECENDING HIERARCHY)

      A) EXPONENTIATION   **
      B) MULTIPLICATION   *
      C) DIVISION         /
      D) ADDITION         +
      E) SUBTRACTION      -

6. EXPRESSIONS :

      A) SINGLE NUMBER OR VARIABLE   EG. 10,-.53,D,X(I,J)

      B) BUILT IN FUNCTION   EG. SQR(10),TAN(.75)

      C) ARITHMETIC EXPRESSION : EXPRESSIONS SEPARATED BY OPERATORS
AND GROUPED BY PARENTHESIS . EG. 5+6.0 , A**2(3*X-4)


go

 7. RELATIONS

      A) GT    A GT B    A GREATER THAN B
      B) GE    A GE B    A GREATER THAN OR EQUAL TO B
      C) LT    A LT B    A LESS THAN B
      D) LE    A LE B    A LESS THAN OR EQUAL TO B
      E) NE    A NE B    A NOT EQUAL TO B
      F)  =    A = B     A.EQUAL TO B


    B. BASIC STATEMENTS

1. REM << ANY SET OF ALPHA-NUMERIC COMMENTS >>

2. READ << VARIABLE,....,VARIABLE >>

3. END


go

 4. LET << VARIABLE >> = << EXPRESSION >>

5. PRINT << 'STRING' OR EXPRESSION,....,'STRING' OR EXPRESSION >>
     OR SIMPLY   PRINT

6. DATA  << 'STRING' OR NUMBER,....,'STRING' OR NUMBER >>
```

```
7. RESTORE  OR  RESTORE$

8. IF << EXPRESSION >> << RELATION >> THEN << STATEMENT NUMBER >>

9. GO TO << STATEMENT NUMBER >>

10. ON << EXPRESSION >> GO TO << STATEMENT NUMBER,....,STATEMENT NUMBER >>

11. FOR << SIMPLE VARIABLE >> = << EXPRESSION >>
                              TO << EXPRESSION >> STEP << EXPRESSION >>
      ....
      ...
     ...
NEXT << SIMPLE VARIABLE >>

12. DIM << LETTER >> ( << INTEGER EXPRESSION >> )       OR
    DIM << LETTER >> ( << INTEGER EXPRESSION,INTEGER EXPRESSION >> )

13. GOSUB << STATEMENT NUMBER >>
      ...
      ...
      ...
RETURN
```

go

```
 THAT CONCLUDES YOUR INSTRUCTION WITH CAI-BASIC .
HOWEVER , YOU ARE INVITED TO USE WHATEVER FACILITIES OF CAI-BASIC
YOU DESIRE AT ANY TIME . IF YOU HAVE NOT RUN PROGRAMS UNDER
THE OS/BATCH MODE (PUNCHING YOUR OWN CARDS AND HANDING THEM
ACROSS THE COUNTER TO BE RUN), YOU SHOULD GET THE BASIC MANUAL,
TECHNICAL NOTE NR. 0211-12 , IN ROOM I-147 TO FIND THE PROPER
JOB CONTROL CARDS REQUIRED . GOOD LUCK WITH THE COMPUTER , AND
REMEMBER IT ONLY DOES WHAT YOU TELL IT TO DO - GIGO (GARBAGE IN
GARBAGE OUT ).


          DO YU WANT TO TERMINATE YOUR INSTRUCTION SESSION ?
 (REPLY : YES OR NO )
```

no

```
          IF YOU WANT TO EXECUTE PROGRAMS AT THIS TIME
THEN REPLY : YES ; OTHERWISE REPLY : NO AND THE INSTRUCTION
WILL CONTINUE .
```

yes

```
               ****  CAIBASIC COMPILER  ****

THE CAIBASIC COMPILER IS A LINE BY LINE INTERPRETER .THE BASIC
```

```
INTERPRETER ACCEPTS STANDARD BASIC STATEMENTS , AND IT ANALYZES
EACH BASIC STATEMENT AS IT IS INPUT . AN ADDED FEATURE OF
THE CAIBASIC COMPILER IS AN EDITING AND A DEBUGGING ROUTINE
THAT ALLOWS THE USER TO ADD , DELETE AND CORRECT STATEMENTS ;
AND TO GET A LISTING OF ALPHA-NUMERIC AND NUMERIC DATA USED ,
AND A TRACE OF ALL SIMPLE VARIABLES AS THEY ARE ASSIGNED VALUES
IN THE PROGRAM.
          THE DEBUG FEATURE  IS USED BY ADDING THE KEY WORD  DEBUG
AS A STATEMENT TO YOUR PROGRAM .
          THE EDITING MODE IS AVAILABLE TO THE USER WHEN AN EXECUTION
ERROR OCCURS AND AFTER SUCCESSFUL EXECUTION .
       IN THE EVENT OF AN INPUT ERROR THE INTERPRETER WILL ANALYZE
 THE ERROR AND PRINT AN ERROR MESSAGE. IF THIS OCCURS FIND THE
ERROR ,AND INPUT THE CORRECT BASIC STATEMENT FOR THE CURRENT
LINE OF INPUT .

   (NOTE : TYPING ERRORS CAN BE DELETED BY TYPING FOUR DOLLAR
SIGNS ($$$$) ON THE SAME LINE AS THE ERROR , HITTING CARRIAGE
RETURN , AND INPUTTING THE LINE AGAIN .)


           **** CAIBASIC EXECUTION ****


      INPUT BASIC PROGRAM NOW (ONE LINE AT A TIME )


let x=1
gosub 100
print x
go to 9999
   100 let x=x+1
       if x gt 3 then 150
       gosub 100
   150 return
9999 end

 4


          DO YOU WANT TO EDIT YOUR PROGRAM
  REPLY : YES  OR NO .

yes
```

```
          **** CAIBASIC EDIT MODE ****

BY USING THE REFERENCE NUMBERS LISTED TO THE LEFT OF YOUR BASIC
PROGRAM STATEMENTS YOU MAY ADD , DELETE , OR CORRECT ONE LINE
OF THE PROGRAM AT A TIME . IN ALL EDITING THE FIRST STEP IS TO
INPUT THE PROGRAM STATEMENT REFERENCE NUMBER AND HIT THE CARRIAGE
RETURN . THE SECOND STEP DEPENDS ON WHAT EDITING YOU DO :

   1. DELETE
THE BASIC STATEMENT REFERENCED IS DELETED BY TYPING THE
LETTERS  DEL
   2. CORRECT
TO CORRECT THE BASIC STATEMENT REFERENCED TYPE IN THE COMPLETE
CORRECT BASIC STATEMENT .
   3. ADD
A BASIC STATEMENT IS ADDED 'AFTER' THE BASIC STATEMENT
REFERENCED BY TYPING THE LETTERS ADD1 FOLLOWED BY THE BASIC
STATEMENT . ALL BLANKS FOLLOWING THE LETTERS  ADD1 WILL BE
INCLUDED IN THE BASIC STATEMENT . TO PLACE A STATEMENT 'BEFORE'
THE FIRST STATEMENT IN THE PROGRAM , USE THE REFERENCE NUMBER 0 .


   1          LET X=1
   2          GOSUB 100
   3          PRINT X
   4          GO TO 9999
   5             100 LET X=X+1
   6                 IF X GT 3 THEN 150
   7                 GOSUB 100
   8             150 RETURN
   9          9999 END

 INPUT REFERENCE NUMBER NOW
8

 INPUT EDITING NOW (DEL,BASIC STATEMENT,ADD1....)


   150 let x=x+1

          MORE CORRECTIONS TO BE MADE ??  REPLY: YES  ;
OR REPLY: NO  AND THE EDITED PROGRAM WILL BE EXECUTED .


yes
```

```
   1          LET X=1
   2          GOSUB 100
   3          PRINT X
   4          GO TO 9999
   5             100 LET X=X+1
   6                 IF X GT 3 THEN 150
   7                 GOSUB 100
   8             150 LET X=X+1
   9          9999 END

 INPUT REFERENCE NUMBER NOW
8

 INPUT EDITING NOW (DEL,BASIC STATEMENT,ADD1....)


add1   return

          MORE CORRECTIONS TO BE MADE ??  REPLY: YES  ;
OR REPLY: NO  AND THE EDITED PROGRAM WILL BE EXECUTED .


no
```

```
     LET X=1
     GOSUB 100
     PRINT X
     GO TO 9999
        100 LET X=X+1
            IF X GT 3 THEN 150
            GOSUB 100
        150 LET X=X+1
        RETURN
     9999 END

 7
```

```
          DO YOU WANT TO EXECUTE ANOTHER PROGRAM
REPLY : YES  OR  NO .

no

          DO YOU WANT TO TERMINATE YOUR INSTRUCTION SESSION ?
 (REPLY : YES OR NO )

yes
R; T=6.99/23.96 15.56.04
```

BIBLIOGRAPHY

1. FENICHEL, R. R., WEIZENBAUM, J., and YOCHELSON, J.C., "Program to Teach Programming," COMMUNICATIONS OF THE ACM, v. 13, March 1971.

2. FENICHEL, R. R., "The Teach System" (unpublished work, PROJECT MAC M-388, MIT, Cambridge, Mass., 1969).

3. HARE, V. C., BASIC PROGRAMMING, Harcourt, Brace, and World, Inc., 1970.

4. KERR, E. G., TING, E. G., and WALDEN, W. E., "A Control Program for Computer Assisted Instruction," 24TH CONFERENCE PROCEEDINGS OF THE ACM, 1969.

5. NOLAN, R. L., INTRODUCTION TO COMPUTING THROUGH THE BASIC LANGUAGE, Holt, Rinehart, and Winston, Inc., 1969.

6. OETTENGER, A. G., RUN, COMPUTER, RUN, Harvard University Press, 1969.

7. SMITH, W. R., and YOUNG, J. L., "Computer Assisted Instruction," MANAGEMENT JOURNAL, Naval Postgraduate School, March 1971.

8. STOLUROW, L. M., "Computer Assisted Instruction," REPORT OF A CONFERENCE, U. S. Department of HEW, Washington, 1969.

9. U. S. COMMISSION ON INSTRUCTIONAL TECHNOLOGY, "To Improve Learning," U. S. Government Printing Office, Washington, 1970.

10. U. S. NAVAL POSTGRADUATE SCHOOL TECHNICAL NOTE NR. 0211-12 "Basic Language Manual," April 1971.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	ENS Alan B. Roberts, Code 53Ro Naval Postgraduate School Monterey, California 93940	1
4.	LT Thomas A. Barry, USN 2730 Murray Ave. Norfolk, VA 23518	1
5.	Assoc Professor Robert C. Bolles, Code 53 Bq Naval Postgraduate School Monterey, California 93940	1

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

ORIGINATING ACTIVITY (Corporate author)

Naval Postgraduate School
Monterey, California 93940

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

REPORT TITLE

CAI-BASIC
A program to Teach the Programming Language "BASIC"

DESCRIPTIVE NOTES (Type of report and inclusive dates)

Master's Thesis; September 1971

AUTHOR(S) (First name, middle initial, last name)

Thomas Anthony Barry

REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
September 1971		10

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

This paper presents a computer aided instruction program that fulfills the objectives of teaching a simple programming language, interpreting student responses, and executing and editing student programs. The CAI-BASIC program is written in FORTRAN IV, level G, and executes on IBM-2741 terminals while running under the CP-67/CMS time sharing system on the U.S. Naval Postgraduate School's IBM-360/67 computer system. The instructional phase of CAI-BASIC presents the fundamentals of "BASIC," a simple user oriented language, in seven lessons. During the instructional sessions the student is presented material and, based on his response to questions, he is routed to the next sequence of instructions. The execution phase of CAI-BASIC allows execution of "BASIC" programs, and has an optional debug feature that provides a trace of program variables to aid the student in finding programming errors. In the event of programming errors the user may enter an edit mode to correct mistakes in his program.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6811

Security Classification

A-31408

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Computer aided instruction						
Computer assisted instruction						
Programmed instruction						

22 FEB 78
22477
25507

Thesis
B24222
c.1
Barry
CAI-BASIC; a program to teach the programming language "BASIC".
130779
BINDERY
22477
25507
22 FEB 78

Thesis
B24222
c.1
Barry
CAI-BASIC; a program to teach the programming language "BASIC".
130779

thesB24222
CAI-BASIC :
3 2768 002 01476 3
DUDLEY KNOX LIBRARY